(1)

PII Redacted

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GA/AA/82D-4

ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS FROM TWO ORBITING SENSORS

THESIS

AFIT/GA/AA/82D-4 Donald W. Gross
Captain USAF

AFIT/GA/AA/82D-4

ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS FROM TWO ORBITING SENSORS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
Donald W. Gross, B.S.
Captain USAF

Graduate Astronautical Engineering
December 1982

Acknowledgements

I would like to express my special thanks to my advisor, Dr. William Wiesel for his help while completing this thesis and attending his classes. His knowledge and willingness to help was a great motivation while pursuing this study.

I would also like to thank my typist, Jill Rueger for her thoroughness and responsiveness in typing the manuscript. Her help is deeply appreciated. Lastly, I would like to express my sincere thanks to my wife, Kim for her patience and encouragement during the period of this research.

Contents

List of Figures

List of Tables

Abstract

A technique was developed for estimation of launch vehicle performance parameters. This technique used an inverse covariance or Bayes filter. Both a seven state and an eight state dynamics model were implemented and their performance investigated. Observations consisted of angular, infrared measurements from two orbital sensors. The seven state filter had 3 position, 3 velocity and an acceleration component for its state vector. The acceleration state was modelled as constant between measurement updates. After the addition of a fading memory, the seven state filter showed good performance in estimating a variable acceleration profile. The eight state filter had 3 states each for position and velocity, and seventh and eighth states involving engine exit velocity and propellant mass flow rate. Although the eight state filter had a better model for the acceleration, the filter proved to be unsuccessful in its estimation attempts.

I Introduction

The objective of this research was to develop space-based procedures to estimate performance parameters of launch vehicles. In the past, estimation techniques of position, velocity and performance parameters of launch and reentry vehicles have been developed using ground-based tracking radars as the data source. The measurements available would include range, range rate, azimuth and elevation.

Having a space reconnaissance capability would allow for a new source of observation data. Angular observations of azimuth and elevation from satellites with passive infrared sensors fall into this category. This information could be used in conjunction with ground based data or give valuable data which was unattainable before due to the lack of ground based stations.

This paper addresses the development of an estimator using these satellite observations as data. An inverse covariance or Bayes filter was used in the development of the estimator. Because no range measurements are available from the IR sensors, there may be degradation in the observability of the states. Of particular interest was the ability of the filter to estimate the accelerations of a launch vehicle as well as its position and velocity.

For the problem, both a seven-state and then an eight-state filter model were evaluated for their performance capabilities. Also, two satellite observers were used for data in order to improve observability in the states.

Assumptions:

(1) The data satellites were assumed to be in geosynchronous equatorial orbits.

(2) The accuracy of the infrared sensors was considered to be the same for both elevation and azimuth angle determination.

(3) The acceleration of the launch vehicle was assumed to act along the velocity vector.

(4) A spherical earth was assumed in the filter model.

Sequence of Presentation:

The derivation of the dynamics equations of the filter are presented in Chapter II. Both seven-state and eight-state models are given. Chapter III shows the development of the observation relationships between the heat emitting ballistic missile and the satellite-based infrared sensor. This is followed by the filter development in Chapter IV. Included in the discussion is the development of the filter equations and the Bayes filter algorithm. The testing and results are discussed in Chapter V. Finally, Chapter VI contains conclusions resulting from the development and presents recommendations for further study.

II Problem Dynamics

In the development of dynamics of the filter, a seven state model was used first. The states included three for position, three for velocity and one for acceleration due to thrust or drag. Later, an eight state model was implemented to see if improved performance of the filter could be attained. In this case the acceleration state is replaced by two states, one for estimating the rocket engine's exit velocity and the other for the rocket's relative mass flow rate. The discussion that follows will first address the development of the dynamics equations for the seven state model and will conclude with the differences needed to implement the eight state model.

The Seven State Model

The equations of motion for the launch vehicle were derived from the general two-body equation:

$$\ddot{\bar{r}} + \frac{\mu}{r^3}\bar{r} = 0 \qquad (2\text{-}1)$$

where μ is the gravitational parameter:

$$\mu = GM \qquad (2\text{-}2)$$

A rectangular earth inertial coordinate frame was used, with the z-direction being north. So $\bar{r}$ was defined by,

$$\bar{r} = x\hat{i} + y\hat{j} + z\hat{k} \qquad (2\text{-}3)$$

where $\hat{i}$, $\hat{j}$, and $\hat{k}$ are unit vectors along the x, y, and z axes respectively. The acceleration was assumed to a⁻t along the velocity vector so:

$$\bar{a} = a\frac{\bar{v}}{|\bar{v}|} \tag{2-4}$$

Adding this to the two body equation gave:

$$\ddot{\bar{r}} = -\frac{\mu}{r^3}\bar{r} + a\frac{\bar{v}}{|\bar{v}|} \tag{2-5}$$

For the seven-state model the state vector was defined by:

$$\bar{x} = \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \\ a \end{bmatrix} \tag{2-6}$$

The system was nonlinear, so the state vector was propagated in time by use of the differential equation:

$$\frac{d}{dt}\bar{x}(t) = \bar{F}(\bar{x}(t), t) \tag{2-7}$$

where $\bar{x}$ is the state vector at time t and $\bar{F}$ is a vector of nonlinear functions of the variables of which $\bar{x}$ is comprised and possibly of t as well.

The three position states are denoted by x, y, and z. The time rates of change of these states are given by their respective velocities:

$$\dot{x} = v_x \tag{2-7a}$$

$$\dot{y} = v_y \tag{2-7b}$$

$$\dot{z} = v_z \tag{2-7c}$$

For the velocity states, the time rates of change were derived by breaking up eq 2-5 into the respective i, j, and k components.

$$\dot{v}_x = - \frac{\mu}{r^3} x + a\frac{v_x}{|\bar{v}|} \tag{2-8a}$$

$$\dot{v}_y = - \frac{\mu}{r^3} y + a\frac{v_y}{|\bar{v}|} \tag{2-8b}$$

$$\dot{v}_z = - \frac{\mu}{r^3} z + a\frac{v_z}{|\bar{v}|} \tag{2-8c}$$

The seventh state, acceleration, was mc lelled as being constant over the time intervals between data updates. Ideally, this would give the model the ability to estimate the deceleration of a reentry vehicle as well as acceleration of a launch vehicle. So the rate of change of acceleration was given by:

$$\dot{a} = 0 \tag{2-9}$$

Then forming $\dot{\bar{x}}$ by use of eqns 2-7, 2-8, and 2-9, gave the $\bar{F}$ vector as:

$$\bar{F} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \\ \dot{a} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ -\frac{\mu}{r^3} x + a\frac{v_x}{|\bar{v}|} \\ -\frac{\mu}{r^3} y + a\frac{v_y}{|\bar{v}|} \\ -\frac{\mu}{r^3} z + a\frac{v_z}{|\bar{v}|} \\ 0 \end{bmatrix} \tag{2-10}$$

For the treatment of nonlinear systems, we assume the availability of a nominal trajectory, $\bar{x}_o(t)$, with a given set of initial conditions $\bar{x}_o(0)$. The true initial conditions differ slightly from the assumed initial conditions by an unknown amount, $\delta\bar{x}(t)$. Assume that the true dynamics solution can be represented by:

$$\bar{x}(t) = \bar{x}_o(t) + \delta\bar{x}(t) \tag{2-11}$$

Differentiation of eq 2-11 gives us:

$$\dot{\bar{x}}(t) = \dot{\bar{x}}_o(t) + \delta\dot{\bar{x}}(t) \tag{2-12}$$

which combining with eq (2-7) yields:

$$\dot{\bar{x}}_o(t) + \delta\dot{\bar{x}}(t) = \bar{F}(\bar{x}_o(t) + \delta\bar{x}(t), t) \tag{2-13}$$

By applying Taylor's theorem and expanding the right side of eq 2-13 we can obtain:

$$\dot{\bar{x}}_o(t) + \delta\dot{\bar{x}}(t) = \bar{F}(x_o(t),\ t) + A(t)\Big|_{\bar{x}_o(t)} \delta\bar{x}(t) + \text{H.O.T.} \tag{2-14}$$

where $\underline{A}(t)$ is a matrix found by taking the partial derivatives of the $\bar{F}$ vector with respect to the state variables, evaluated at the nominal solution, $\bar{x}_o(t)$.

$$\underline{A}(t)\Big|_{\bar{x}_o(t)} = \frac{\partial\bar{F}}{\partial\bar{x}}\Big|_{\bar{x}_o(t)} \tag{2-15}$$

With the assumption that $\delta\bar{x}$ is small, the nominal trajectory satisfies the dynamics model:

$$\dot{\bar{x}}_o(t) = \bar{F}(\bar{x}_o(t),\ t) \tag{2-16}$$

Subtracting this from eq 2-14 and neglecting higher order terms, we see that to first order, $\delta\bar{x}(t)$ satisfies the time-varying linear differential equaticn:

$$\delta\dot{\bar{x}}(t) = \underline{A}(t)\Big|_{\bar{x}_o(t)} \delta\bar{x}(t) \tag{2-17}$$

The derivation of the $\underline{A}(t)$ matrix is given in Appendix A.

The system given by eq 2-17 is linear and time dependent. Therefore, the variations in the state are propagated by the state transition matrix, $\underline{\Phi}$:

$$\delta\bar{x}(t) = \underline{\Phi}(t,t_o)\,\delta\bar{x}(t_o) \tag{2-18}$$

The state transition matrix is obtained by the solution of:

$$\dot{\underline{\Phi}}(t,t_o) = \underline{A}(t)\Big|_{\bar{x}_o(t)} \underline{\Phi}(t,t_o) \tag{2-19}$$

with the initial conditions:

$$\underline{\Phi}(t_o, t_o) = \underline{I} \tag{2-20}$$

where $\underline{I}$ is the identity matrix.

The Eight-State Model

For the derivation of the eight state model, thrust was assumed to be constant for each stage of the launch vehicle. The dynamics of the system remained basically the same with only the $\overline{F}$ vector and $\underline{A}(t)$ matrix changing. With thrust constant, the acceleration can be given by:

$$a = v_e \frac{\dot{m}}{(m_o - \dot{m}t)} \tag{2-21}$$

where:

a = acceleration due to thrust

$\dot{m}$ = the propellant mass flow rate of the rocket engine

m_o = the original mass of the launch vehicle (including propellant)

t = time

By using eq (2-21) in this form, we would need three new states—v_e, $\dot{m}$, and m_o. However, by dividing the numerator and denominator by m_o, the acceleration becomes:

$$a = v_e \frac{\frac{\dot{m}}{m_o}}{(1 - \frac{\dot{m}}{m_o}t)} = v_e \frac{M}{(1 - Mt)} \tag{2-22}$$

where M is the relative mass flow rate. The two additional states become v_e and M. The state vector for the eight state model is:

$$\bar{x} = \begin{bmatrix} x \\ y \\ z \\ v_x \\ v_y \\ v_z \\ v_e \\ M \end{bmatrix} \tag{2-23}$$

and the resulting $\bar{F}$ vector becomes:

$$\dot{\bar{x}} = \bar{F}(\bar{x}(t), t) = \begin{bmatrix} v_x \\ v_y \\ v_z \\ -\frac{\mu}{r^3} x + a\frac{v_x}{|\bar{v}|} \\ -\frac{\mu}{r^3} y + a\frac{v_y}{|\bar{v}|} \\ -\frac{\mu}{r^3} z + a\frac{v_z}{|\bar{v}|} \\ 0 \\ 0 \end{bmatrix} \tag{2-24}$$

The changes in the $\underline{A}(t)$ matrix are given in Appendix A.

Both state models were investigated in the implementation of the estimator. Although not physically correct, the seven-state model was attractive because of its general nature. This would hopefully allow the filter to adapt to sudden changes in acceleration, such as those experienced in a rocket staging. However, with the acceleration assumed constant, one would have to be careful on how many data points were used in an update. If the change in acceleration was great in a data span, this would make the filter's constant acceleration approximation less valid. The eight state model has a better physical representation of the acceleration state. This should lend to batching a larger amount of data to get an improved update. But, the ability of the filter to detect staging may be degraded. The characteristics of each model will be discussed in more detail later.

III Observation Relationships

Azimuth and Elevation Derivation

The derivation of the angular relationships for the problem was obtained from Miller's presentation (Ref 6). The following will basically follow his development of the observation angles.

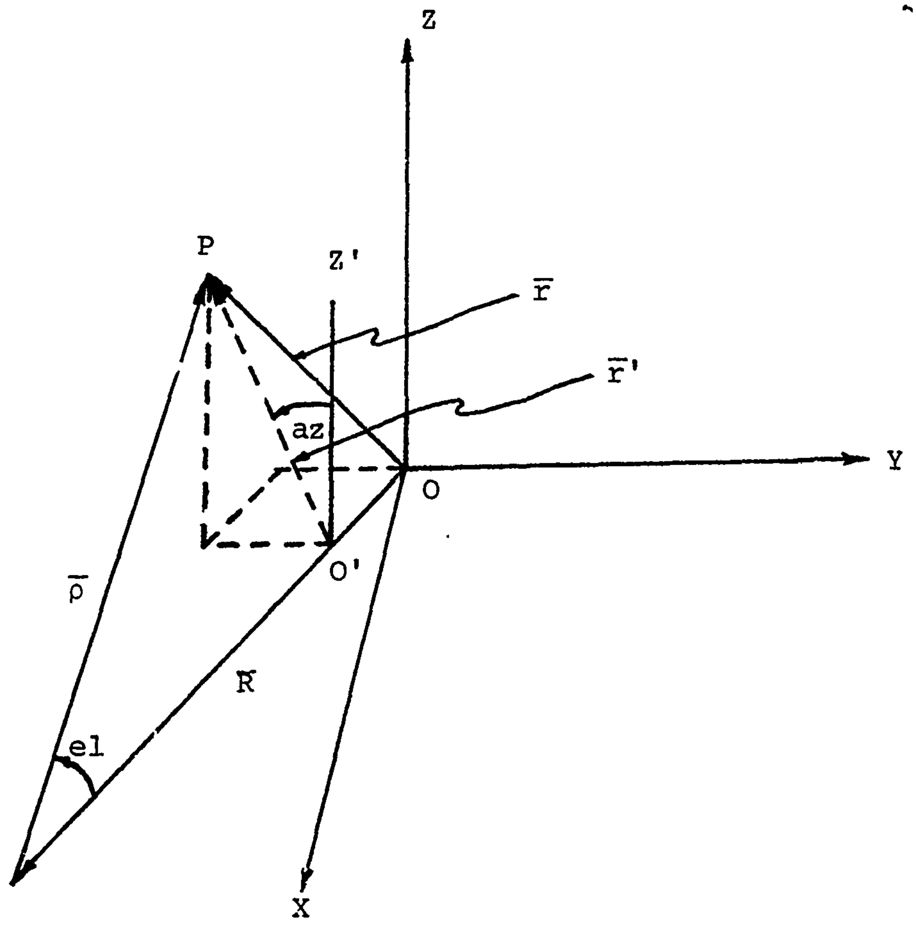
The measurements from the observation satellite were the two angles, azimuth and elevation, depicted in Figure 1. These angular relationships were derived with the observer and launch vehicle in the same rectangular coordinate frame. Azimuth was the angle (clockwise direction being positive) between a local vertical, z', and a local position vector, $\bar{r}'$. Elevation was the angle between the negative of the position vector of the observer, $-\bar{R}$, and the vector from the observer to the target vehicle, $\bar{\rho}$. Both angles were measured in radians. The angular relationships were derived from the equations

$$\cos \gamma = \frac{\bar{a} \cdot \bar{b}}{|\bar{a}|\,|\bar{b}|} \tag{3-1}$$

$$\sin \gamma = \frac{|\bar{a} \times \bar{b}|}{|\bar{a}|\,|\bar{b}|} \tag{3-2}$$

Elevation is the angle between $\bar{\rho}$ and $-\bar{R}$, so from Figure 2 and eq 3-1 we see:

$$\cos \phi = \frac{|\bar{s}|}{|\bar{r}|} = \frac{\bar{s} \cdot \bar{r}}{|\bar{s}| \cdot |\bar{r}|} = \frac{\bar{R} \cdot \bar{r}}{|\bar{R}|\,|\bar{r}|} \tag{3-3}$$



$\overline{R}$ = position vector of observer

$\overline{r}$ = position vector of launch vehicle

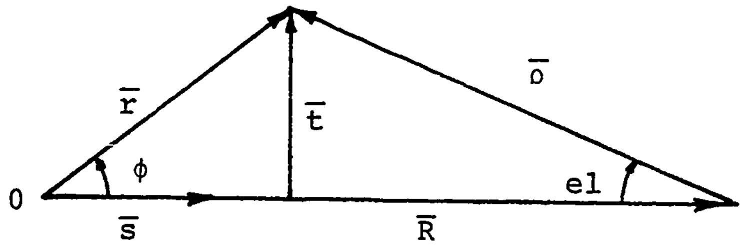
$\overline{\rho}$ = position vector of launch vehicle relative to observer

$\overline{r}'$ = position vector (in a plane orthogonal to $\overline{R}$) of launch vehicle relative to 0'

el = elevation angle; the angle subtended by $\overline{\rho}$ and $-\overline{R}$

az = azimuth angle; the angle subtended by $\overline{r}'$ and the line segment from 0' to z'

Figure 1. Illustration of observation angles



$\bar{t}$ = a unit vector normal to $\bar{R}$ and containing point P

$\bar{s}$ = a unit vector along $\bar{R}$

Figure 2. Geometry for elevation derivation

From eq 3-3

$$|\bar{s}| = |\bar{r}| \cos \phi \tag{3-4}$$

Now introducing direction:

$$\bar{s} = \frac{\bar{R}}{|\bar{R}|} |\bar{r}| \cos \phi \tag{3-5}$$

Using vector addition we get:

$$\bar{s} + \bar{t} = \bar{r} \tag{3-6}$$

and substituting into eq 3-5 gives:

$$\frac{\bar{R}}{|\bar{R}|} |\bar{r}| \cos \phi + \bar{t} = \bar{r} \tag{3-7}$$

By rearranging:

$$\bar{t} = \bar{r} - \frac{\bar{R}}{|\bar{R}|}\,|\bar{r}|\,\cos\phi \tag{3-8}$$

and combining with the results of eq 3-3 we get:

$$\bar{t} = \bar{r} - \frac{\bar{R}}{|\bar{R}|}\,|\bar{r}|\left[\frac{\bar{R}\cdot\bar{r}}{|\bar{R}|\,|\bar{r}|}\right] = \bar{r} - \bar{R}\left[\frac{\bar{R}\cdot\bar{r}}{|R|^2}\right] \tag{3-9}$$

From Figure 2 we see:

$$\sin(\mathrm{el}) = \frac{|\bar{t}|}{|\bar{\rho}|} \tag{3-10}$$

and:

$$\bar{\rho} = \bar{r} - \bar{R} \tag{3-11}$$

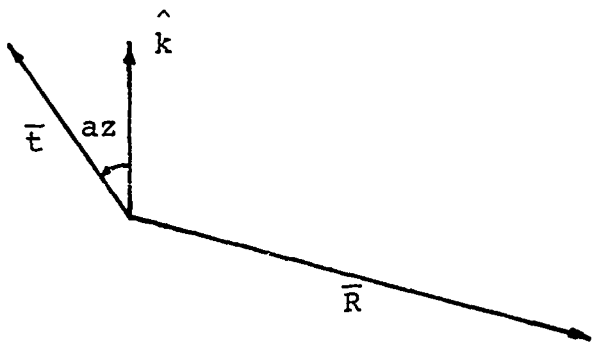
Now substituting eqs 3-9 and 3-11 into eq 3-10 we get:

$$\sin(\mathrm{el}) = \frac{\left[\bar{r} - \bar{R}\left(\frac{\bar{R}\cdot\bar{r}}{|\bar{R}|^2}\right)\right]}{|\bar{r} - \bar{R}|} \tag{3-12}$$

which gives us the relationship for the elevation angle as:

$$\mathrm{el} = \sin^{-1}\left[\frac{\bar{r} - \bar{R}\left(\frac{\bar{R}\cdot\bar{r}}{|\bar{R}|^2}\right)}{|\bar{r} - \bar{R}|}\right] \tag{3-13}$$

Azimuth was given as the angle between a local vertical and a local position vector. So from Figure 3 and using eq 3-1 we get:



$\bar{t}$ = vector the same as in Figure 2; normal to $\bar{R}$ and containing point P.

$\hat{k}$ = a unit vector in the z direction.

Figure 3. Geometry for azimuth derivation

$$\cos(az) = \frac{\bar{t} \cdot \hat{k}}{|\bar{t}|\,|\hat{k}|} = \frac{\bar{t} \cdot \hat{k}}{|\bar{t}|} \qquad (3\text{-}14)$$

and substituting eq 3-9 in for $\bar{t}$ gives:

$$\cos(az) = \frac{\left[\bar{r} - \bar{R}\left(\frac{\bar{R} \cdot \bar{r}}{|\bar{R}|^2}\right)\right] \cdot \hat{k}}{\left|r - R\left(\frac{\bar{R} \cdot \bar{r}}{|\bar{R}|^2}\right)\right|} \qquad (3\text{-}15)$$

This gives us the relationship for the azimuth angle as:

$$az = \cos^{-1}\left[\frac{\left[\bar{r} - \bar{R}\left(\frac{\bar{R} \cdot \bar{r}}{|\bar{R}|^2}\right)\right] \cdot \hat{k}}{\left|\bar{r} - \bar{R}\left(\frac{\bar{R} \cdot \bar{r}}{|\bar{R}|^2}\right)\right|}\right] \qquad (3\text{-}16)$$

From eq 3-15, the sign of the azimuth angle is not readily discernable. So to determine the sign, we look at the z component of the cross product of $\bar{R} \times \bar{r}$. If the z component was negative, then the azimuth was negative.

Observation Relation Derivation

The relationships of the angular measurements, azimuth and elevation, are nonlinear functions of the state vector. A set of discrete observations are related to the state variables by the general nonlinear relationship:

$$\bar{z}(t_i) = \bar{G}(\bar{x}(t_i),t_i) \tag{3-17}$$

Evaluating the observation vector along the nominal trajectory, $\bar{x}_o(t)$, at discrete times, t_i, yields the nominal measurements:

$$\bar{z}_o(t_i) = \bar{G}(\bar{x}_o(t_i),t_i) \tag{3-18}$$

Now the true observations satisfy the equation:

$$\bar{z}(t_i) = \bar{G}(\bar{x}_o(t_i) + \delta\bar{x}(t_i),t_i) \tag{3-19}$$

By expanding this equation in a Taylor's series about the nominal trajectory yields:

$$\bar{z}(t_i) = \bar{G}(\bar{x}_o(t_i),t_i) + \left.\frac{\partial\bar{G}(\bar{x}_o(t_i),t_i)}{\partial\bar{x}(t_i)}\right|_{\bar{x}_o(t_i)} \delta\bar{x}(t_i) + \text{H.O.T.} \tag{3-20}$$

subtracting eq 3-18 from both sides of eq 3-20 and neglecting higher order terms gives a relationship for the residuals of the observations, $\bar{r}(t_i)$—the difference between the true and the nominal observations:

$$\bar{r}(t_i) \cong \bar{z}(t_i) - \bar{z}_o(t_i) = \left.\frac{\partial \bar{G}}{\partial \bar{x}}\right|_{\bar{x}_o(t_i)} \delta\bar{x}(t_i) \qquad (3\text{-}21a)$$

$$= \underline{H}(\bar{x}_o(t_i),t_i)\,\delta\bar{x}(t_i) \qquad (3\text{-}21b)$$

where $\bar{H}$ is the partial derivative matrix of the observation vector with respect to the state vector evaluated at the nominal trajectory.

To make the residual the difference at the epoch, t_o instead of t_i, the state transition matrix is once again used:

$$\bar{r}(t_i) \cong \underline{H}(\bar{x}_o(t_i),t_i)\underline{\Phi}(t_i,t_o)\,\delta\bar{x}(t_o) \qquad (3\text{-}22a)$$

$$\cong \underline{T}(t_i)\,\delta\bar{x}(t_o) \qquad (3\text{-}22b)$$

The elements of the $\underline{H}$ matrix are derived in Appendix B.

IV Filter Development

This chapter shows the derivation of the equations for the Bayes filter and the algorithm for implementation of the filter.

The Bayes Filter

From Chapter II, the problem dynamics were given by the vector equation:

$$\dot{\bar{x}} = \bar{F}(\bar{x},t) \qquad (2\text{-}7)$$

Also, it was shown that with the dynamics of the problem well understood, deviations in the state could be expressed by:

$$\delta\bar{x}(t) = \underline{\Phi}(t,t_o)\delta\bar{x}(t_o) \qquad (2\text{-}18)$$

as long as the $\delta\bar{x}$'s were small.

In Chapter III, a relationship for the residual, $\bar{r}(t_i)$, was derived by finding the observation relation $\underline{H}(t_i)$ and using eq 2-18 and was given by:

$$\bar{r}(t_i) \cong \underline{H}(t_i)\underline{\Phi}(t_i,t_o)\delta\bar{x}(t_o)$$

$$\cong \underline{T}(t_i)\delta\bar{x}(t_o) \qquad (3\text{-}22)$$

Due to corruptive noise in the measurements, there is an error in the residual approximation. This is shown as:

$$\bar{r}(t_i) = \underline{T}(t_i)\delta\bar{x}(t_o) + \bar{e}(t_i) \qquad (4\text{-}1)$$

Define $\underline{Q}_i$ as the observation covariance matrix, containing information about the accuracy of the data. For this application

$\underline{Q}_i$ is chosen as diagonal under the conditions that the random errors in azimuth and elevation are uncorrelated within a given observation.

Now, using Gaussian error statistics, the probability density function for the error vector is expressed as:

$$f(\bar{e}) = (2\pi)^{-N/2}|\underline{Q}|^{-\frac{1}{2}}\exp(-\tfrac{1}{2}J) \tag{4-2}$$

where $J = \bar{e}^T\underline{Q}^{-1}\bar{e}$.

To maximize the probability, the principle of maximum likelihood is used. J, the weighted least squares cost function, must be minimized in order that f be a maximum. J is of the form:

$$J = [\bar{r}(t_i) - \underline{T}(t_i)\delta\bar{x}(t_o)]^T\underline{Q}_i^{-1}[\bar{r}(t_i) - \underline{T}(t_i)\delta\bar{x}(t_o)] \tag{4-3}$$

Since $\underline{Q}_i^{-1}$ is positive definite, the necessary and sufficient condition for minimizing J is:

$$\frac{\partial J}{\partial\delta\bar{x}(t_o)} = \bar{0}^T \tag{4-4}$$

solving for $\delta\bar{x}(t_o)$ yields (ref 5):

$$\delta\bar{x}(t_o) = (\underline{T}(t_i)^T\underline{Q}_i^{-1}\underline{T}(t_i))^{-1}\underline{T}(t_i)^T\underline{Q}_i^{-1}\bar{r}(t_i) \tag{4-5}$$

With $\delta\bar{x}$ assumed as zero mean, the covariance associated with $\delta\bar{x}$ is:

$$\underline{P}_{\hat{x}}(t_o) = E(\delta\bar{x}(t_o)\delta\bar{x}(t_o)^T) \tag{4-6}$$

where E is the expectation operator. Let $\underline{W} = (\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1}\underline{T}^T\underline{Q}^{-1}$ and substituting $\delta\bar{x}$ from eq 4-5 gives:

$$\underline{P}_{\hat{x}}(t_o) = E(\underline{W}\,\bar{r}\,\bar{r}^T\,\underline{W}^T) \tag{4-7}$$

and since $\underline{W}$ is deterministic:

$$\underline{P}_{\hat{x}}(t_o) = \underline{W}\,E(\bar{r}\,\bar{r}^T)\,\underline{W}^T \tag{4-8}$$

But the expected value of $\bar{r}\,\bar{r}^T$ is defined as the covariance of the observations, $\underline{Q}$. So eq 4-8 can be written as:

$$\underline{P}_{\hat{x}}(t_o) = \underline{WQW}^T \tag{4-9}$$

Expanding and simplifying:

$$\begin{aligned}\underline{P}_{\hat{x}}(t_o) &= (\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1}\underline{T}^T\underline{Q}^{-1}\underline{Q}[(\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1}\underline{T}^T\underline{Q}^{-1}]^T\\ &= (\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1}\underline{T}^T(\underline{T}^T\underline{Q}^{-1})^T[(\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1}]^T\end{aligned}$$

$$\underline{P}_{\hat{x}}(t_o) = (\underline{T}^T\underline{Q}^{-1}\underline{T})^{-1} \tag{4-10}$$

Equations 4-5 and 4-10 are used in basic least squares estimation. A sequential manner of handling the data can be used and this is the Bayes filter. This is done by combining an old estimate, $\hat{\bar{x}}^-(t_o)$, and its covariance, $\underline{P}(t_o^-)$, with a new batch of data $\bar{z}$. $\hat{\bar{x}}^-(t_o)$ is treated as data, rather than reprocessing all the old data that went into forming the estimate. So, the observation relations for the new estimate are:

$$\bar{x} = \underline{I}\,\hat{\bar{x}}\,(t_o^-) \tag{4-11}$$

$$\bar{z} = \underline{G}\,(\bar{x}(t_i),t_i)\ . \tag{4-12}$$

The augmented matrices become

$$\underline{T}_{aug} = \begin{bmatrix} \underline{I} \\ \underline{T}_i \end{bmatrix} \qquad (4\text{-}13)$$

$$\underline{Q}_{aug} = \begin{bmatrix} \underline{P}(-) & \underline{0} \\ \underline{0} & \underline{Q}_i \end{bmatrix} \qquad (4\text{-}14)$$

$$\bar{r}_{aug} = \begin{bmatrix} (\hat{\bar{x}}^- - \bar{x}_o) \\ \bar{r}_i \end{bmatrix} \qquad (4\text{-}15)$$

where $\bar{x}_o$ is the assumed nominal state.

With this, the new covariance is:

$$\begin{aligned} \underline{P}^+ &= (\underline{T}^T_{aug}\ \underline{Q}^{-1}_{aug}\ \underline{T}_{aug})^{-1} \\ &= (\underline{P}^{-1}(-) + \underline{T}^T_i\ \underline{Q}^{-1}_i\ \underline{T}_i)^{-1} \qquad (4\text{-}16) \end{aligned}$$

and the correction to the state vector is:

$$\begin{aligned} \delta\bar{x} &= (\underline{T}^T{}_{aug}\ \underline{Q}^{-1}_{aug}\ \underline{T}_{aug})^{-1}\ \underline{T}^T_{aug}\ \underline{Q}^{-1}_{aug}\ \bar{\underline{r}}_{aug} \\ &= (\underline{P}^{-1}(-) + \underline{T}^T_i\ \underline{Q}^{-1}_i\ \underline{T}_i)^{-1}\ (\underline{P}^{-1}(-)(\hat{\bar{x}} - \bar{x}_o) + \underline{T}^T_i\ \underline{Q}^{-1}_i\ \bar{r}_i) \end{aligned}$$

The algorithm below shows the step by step iterative process used to converge upon a solution.

<u>The Bayes Filter Algorithm</u>

1. Input

    a. Estimate at epoch

        1. $\hat{\bar{x}}(t_o)$

2. $\underline{P}(t_o^-)$

b. New Data

1. $\bar{z}_i$

2. $\bar{Q}_i$

2. Using a nominal solucion $\bar{x}_o(t_o)$, integrate dynamics to obtain:

a. $\bar{x}_o(t_i)$

b. $\underline{\Phi}(t_i, t_o)$

3. From each measurement calculate:

a. $\bar{r}_i = \bar{z}_i - \underline{G}\ (\bar{x}_o(t_i), t_i)$

b. $\underline{H}_i$

4. Assemble vector/matrices necessary for filter equations

$$\bar{r} = \begin{bmatrix} \bar{r}_1 \\ \bar{r}_2 \\ \cdot \\ \cdot \\ \cdot \\ \bar{r}_i \end{bmatrix}, \quad \underline{T} = \begin{bmatrix} \underline{H}_1\ \underline{\Phi}_1 \\ \underline{H}_2\ \underline{\Phi}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{H}_i\ \underline{\Phi}_i \end{bmatrix}, \quad \underline{Q} = \begin{bmatrix} \underline{Q}_1 & & & & \\ & \underline{Q}_2 & & & \\ & & \cdot & & \\ & & & \cdot & \\ & & & & \cdot \\ & & & & \underline{Q}_i \end{bmatrix}$$

5. Compute update of covariance and state:

a. $\underline{P}(+) = (\underline{P}^{-1}(t_i^-) + \underline{T}^T\ \underline{Q}^{-1}\ \underline{T})^{-1}$

b. $\delta\bar{x} = \underline{P}(+)(\underline{P}^{-1}(t_i^-)(\hat{\bar{x}}(t_i) - \bar{x}_o(t_i)) + \underline{T}^T\ \underline{Q}^{-1}\ \bar{r})$

6. Update the nominal solution:

$$\bar{x}_o(t_i) = \bar{x}_o(t_i) + \delta\bar{x}$$

7. Convergence check. If convergence criteria is met then:

a. $\hat{\bar{x}}(t_i) = \bar{x}_o(t_i)$

b. $\underline{P}(t_i^+) = \underline{P}(+)$

If not met, return to step 2 with newly computed $\bar{x}_o(t_o)$ and repeat the process.

8. Propagate estimate and covariance to new epoch and begin process over again. Continue until launch trajectory is complete. The covariance is propagated by:

$$\underline{P}(t_{i+1}^-) = \underline{\Phi}^T(t_{i+1}, t_i)\underline{P}(t_i^+)\underline{\Phi}(t_{i+1}, t_i)$$

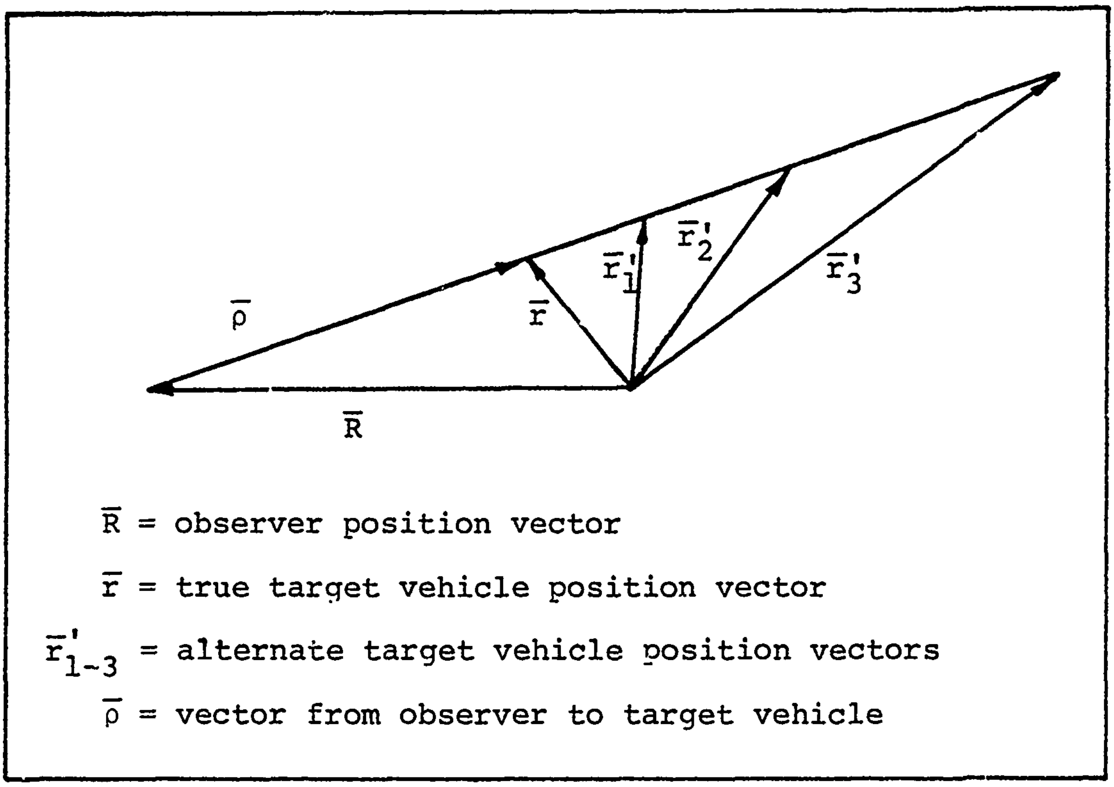
The convergence criteria is based upon the true solution being the result. Theoretically $\delta\bar{x}$ will converge to zero. However, in practice, it should be allowed to converge to within the associated square root of the covariance for that element, $\sqrt{P_{ii}}$. The residuals, independently, should be of order $\sqrt{Q_{ii}}$ as they converge.

Although no estimator is completely self starting, the Bayes filter can be started with only a guess for the nominal state. This is reflected in the algorithm by initializing the inverse of the covariance matrix, $\underline{P}^{-1}(t_o^-)$ as the null matrix. This indicates that there is no a priori knowledge of the system, which can certainly be the case when looking at the launch data of a missile. In this case, the Bayes filter reverts to a least squares filter and the first update is accomplished looking only at the measurements.

V Testing and Results

Setup of Observers

As mentioned before, two observers were used in the problem, assumed to be in geosynchronous equatorial orbit. This stereo view would, hopefully, prevent any problems in observability that might arise using only one observer. These problems could occur if launch was directly below the observer or the launch vehicle was traveling away from the observer. The second situation is displayed in Figure 4.



$\overline{R}$ = observer position vector

$\overline{r}$ = true target vehicle position vector

$\overline{r}'_{1-3}$ = alternate target vehicle position vectors

$\overline{\rho}$ = vector from observer to target vehicle

Figure 4. Measurement ambiguity with one observer

This shows that along certain trajectories there is very little change in the azimuth and elevation measurements. Therefore, not much information would be given to the filter regarding changes in the states, especially velocity and acceleration.

The tracking sequences were initialized with the two observers po:itioned 90° apart along the same orbit. One was positioned on the -y axis and the other on the x axis as shown in Figure 5. This caused no loss in generality since the coordinate system chosen was arbitrary and the launch vehicle was initialized from a point that was not on any of the coordinate axes. During tracking, the observers progressed counterclockwise along their orbit.

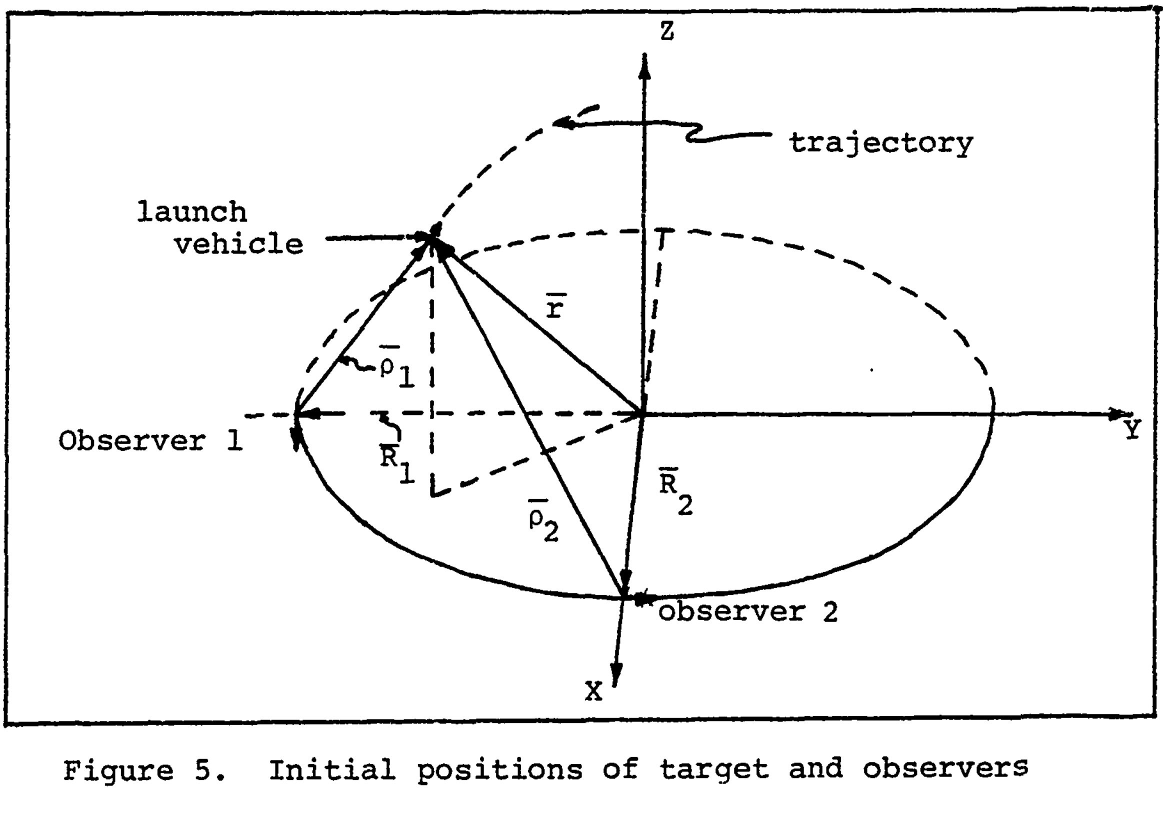


Figure 5. Initial positions of target and observers

Truth Model

The truth model generated the true azimuth and elevation data at given times during the launch profile. This data was generated using a computer program that propagated the states of the launch vehicle using the two body equation of motion. The position and velocity were propagated using the equations previously developed in Chapter II, eqs 2-7 and 2-8. To model the acceleration, it was assumed that the thrust was constant for each stage. With this, the acceleration was given by eq 2-22. Then taking the time derivative gave:

$$\dot{a} = \frac{V_e M^2}{(1 - Mt)^2} \qquad (5\text{-}1)$$

which was used in the truth model to propagate the acceleration state.

The position states were then combined with the position vectors of each observer at the respective times and substituted into eqs 3-13 and 3-16 for the angular measurements required for the data.

Values for the thrust profile were derived from parameters of a Titan IIIB rocket (ref 8).

1st Stage	V_e - 8243.2 ft/sec
	Thrust - 464,900 lb
	Propellant mass flow rate - 56.398 lb-sec/ft
	Initial mass - 11,275 lb-sec^2/ft
2nd Stage	V_e - 10,220.3 ft/sec
	Thrust - 102,300 lb
	Propellant mass flow rate - 10.0095 lb-sec/ft

Initial mass - 2735 lb-sec^2/ft

3rd Stage V_e - 9402.4 ft/sec

Thrust - 16,000 lb

Propellant mass flow rate - 1.7017 lb-sec/ft

Initial mass - 455 lb-sec^2/ft

Figure 6 shows the acceleration profile given by the truth model.

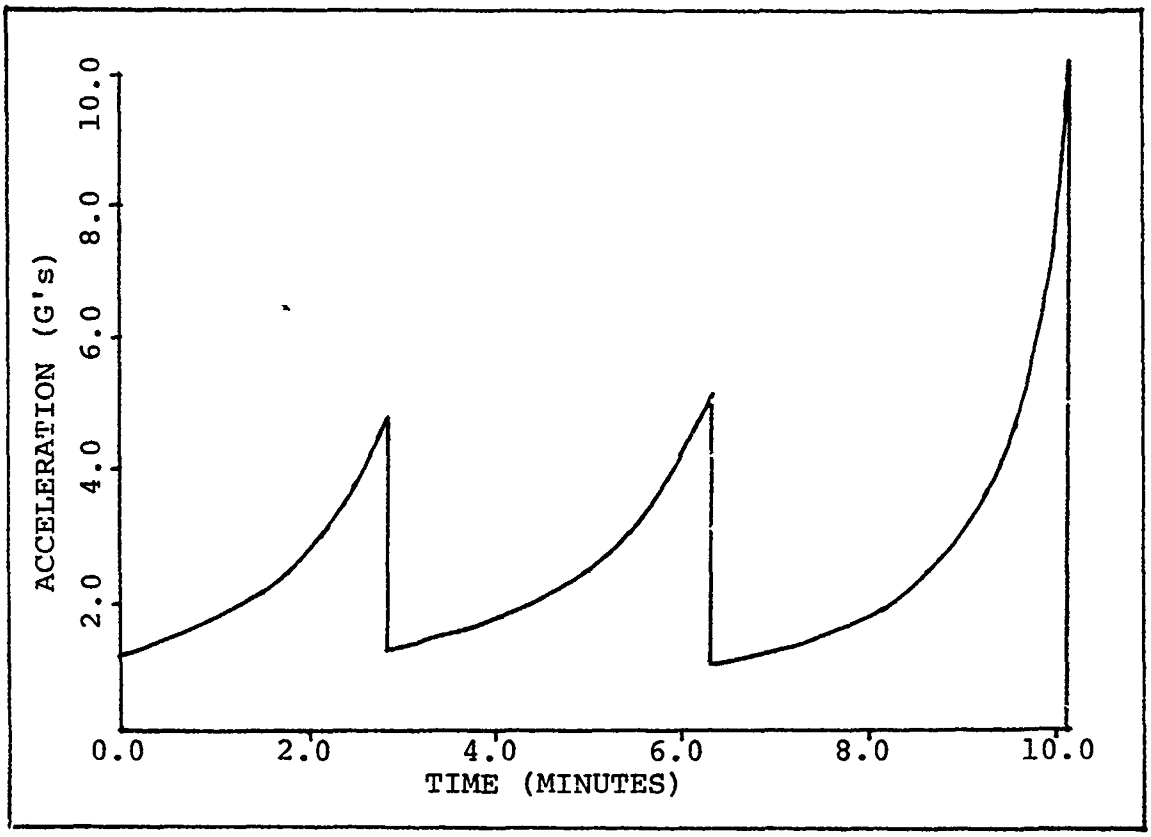


Figure 6. Truth model acceleration versus time

Computer Program Development

The first step in developing the computer program for the filter was verification of the partial matrices $\underline{A}$ and $\underline{H}$ that were developed in Chapters II and III. After setting up the $\underline{A}$ matrix as given in Appendix A, the individual elements were verified by conducting a numerical check given by:

$$A_{ij} \cong \frac{F_i(x_j + \delta,t) - F_i(\overline{x},t)}{\delta} \tag{5-2}$$

where:

A_{ij} = the element in the i^{th} row and j^{th} column

δ = a deviation on the order of 10^{-5} or smaller

$F_i(x,t)$ = the i^{th} component of $\overline{F}$, evaluated at $\overline{x}$

$F_i(x_j + \delta,t)$ = the i^{th} component of F, calculated when δ has been added to the j^{th} state of $\overline{x}$

The check gives an approximation to the elements of $\underline{A}$ as a result of small changes in the state vector, $\overline{x}$. If the matrix, $\underline{A}$, derived numerically agrees with $\underline{A}$ evaluated at the state, $\overline{x}$, this provides assurance that the partial derivatives taken in deriving $\underline{A}$ are correct.

The $\underline{H}$ matrix was set up as given in Appendix B and was verified in a similar manner to the $\underline{A}$ matrix. The verification resulted from looking at the equation:

$$\underline{H}_i \cong \frac{\underline{G}(x_i + \delta,t) - \underline{G}(\overline{x},t)}{\delta} \tag{5-3}$$

where $\underline{H}_i$ = the i^{th} column of the $\underline{H}$ matrix

δ = same as before

$\underline{G}(\overline{x}, t)$ = the evaluation of $\underline{G}$, at the state vector $\overline{x}$

$\underline{G}(x_i + \delta,t)$ = the evaluation of $\underline{G}$ with δ added to the i^{th} state

After the partials in $\underline{A}$ and $\underline{H}$ were all shown as correct, a check on the $\underline{\Phi}$ matrix was performed. Again a numerical check was used to see if $\underline{\Phi}$ was propagated in time correctly. The $\underline{\Phi}$ matrix was propagated using eq 2-19:

$$\dot{\underline{\Phi}}(t,t_o) = \underline{A}(t)\underline{\Phi}(t,t_o) \tag{2-19}$$

and the check was accomplished by using:

$$\underline{\Phi}_i = \frac{\overline{X}(x_i(t_o) + \delta,t) - \overline{X}(\overline{x}(t_o),t)}{\delta} \tag{5-4}$$

where $\underline{\Phi}_i$ = the i^{th} column of the $\underline{\Phi}$ matrix

δ = same as before

$\overline{X}(\overline{x}(t_o),t)$ = the state vector as a function of t and initial conditions, $\overline{x}(t_o)$

$\overline{X}(x_i(t_o) + \delta,t)$ = the state vector as a function of t and initial conditions, $\overline{x}(t_o)$ with δ added to the i^{th} state

With the matrices checked out the next step was to set up and check out the filter's ability to converge to a solution. This was done by setting up a least squares filter. The least squares algorithm is essentially the same as the Bayes, the basic difference being that least squares estimation only minimizes the squares of the measurement residuals and doesn't use information from previous state estimates. So, the update equations for the covariance and state estimate were given by:

$$\underline{P}(+) = [\underline{T}^T(t_i)\underline{Q}_i^{-1}\underline{T}(t_i)]^{-1} \tag{5-5}$$

$$\overline{x} = \underline{P}(+)\underline{T}(t_i)\underline{Q}_i^{-1}\overline{r}(t_i) \tag{5-6}$$

Since the Bayes filter during initial update reverts to least squares estimation, if no a priori information is known, this was a good check to see if the Bayes filter would have any problems with convergence to a solution at epoch.

The check was accomplished using simulated measurements to a satellite in orbit, first with zero acceleration and then with a constant acceleration. For ease of computation, astrodynamic units from Bate, Mueller and White (ref 1:429) were used in all test cases. Five measurements at 0.33 time unit (approximately 266 second) intervals were used in the filter update. Each measurement consisted of an elevation and an azimuth from both observers. The least squares filter was able to detect and correct perturbations in the first six states on the unaccelerated trajectory as shown in Table 1. With a constant acceleration of 1/6 g's, the filter was again able to converge to the correct solution after perturbations in all seven states. This is shown in Table 2.

In both cases, the least squares filter gave an estimate, accurate to seven decimal places of the true solution, within two iterations. Although, in each of the cases the acceleration was constant, the results gave some optimism in that the estimation problem using the infrared sensors may be plausible. It should be noted that the time interval between measurements was quite large. Time intervals of this magnitude are unacceptable for estimating acceleration during the ascent stages of a missile launch. As mentioned before, a large variation in the acceleration over a data span would

Table 1. Good correction to first six states

	Filter $\hat{x}$ initial	$\Delta\bar{x}$ from true	Corrections		
			1st	2nd	3rd
1	1.0290466	.1E-4	-.118312E-4	.182258E-5	.4215E-11
2	.0005	.5E-3	-.495331E-3	-.468180E-5	.1958E-10
3	-.0006	-.6E-3	.598687E-3	.131304E-5	.7479E-10
4	-.0005	-.5E-3	.505352E-3	-.533652E-5	-.2134E-10
5	.9857	-.9E-4	.781742E-4	.122667E-4	-.5657E-10
6	.0005	.5E-3	-.497214E-3	-.278635E-5	-.1401E-9
7	.0	.0	.164532E-4	-.165155E-4	-.7145E-10

Notes: Corrections based on 3 sets of 5 observations
Observations at 0.33 time unit (≈266 sec) intervals
Unaccelerated trajectory

Table 2. Good correction to all seven states

	filter $\hat{x}$ initial	$\Delta\bar{x}$ from true	Corrections		
			1st	2nd	3rd
1	1.0290466	.1E-4	-.130934E-4	.308629E-5	-.9982E-11
2	.0005	.5E-3	-.485662E-3	-.143514E-4	.9722E-10
3	-.0006	-.6E-3	.596834E-3	.316469E-5	.8925E-9
4	-.0005	-.5E-3	.510810E-3	-.108001E-4	-.4067E-10
5	.9857	-.9E-4	.511362E-4	.393112E-4	-.3160E-9
6	.0005	.5E-3	-.494899E-3	-.509932E-5	-.1413E-8
7	.16665	-.167E-4	.685476E-4	-.519284E-4	.4755E-9

Notes: Corrections based on 3 sets of 5 observations
Observations at 0.33 time unit (≈266 sec) intervals
Trajectory acceleration constant 1/6 g

make the constant acceleration approximation less valid. Several data rates were investigated while checking out the Bayes filter.

The Seven State Filter

Knowing that the matrices and all other subroutines were performing correctly, the computer program was set up for the Bayes algorithm given in Chapter IV. The program is presented in Appendix C.

The initial testing of the filter used perfect data at one second intervals. Five sets of measurements were used for each update. A constant thrust launch profile was simulated to determine if the filter could follow a target vehicle through an entire trajectory. With deviations in the initial guess for the first and seventh states, the filter was able to converge to a correct solution within three iterations and then estimate all the states to within six decimal places throughout the launch.

The next test was to determine the filter's ability to estimate a variable acceleration profile. Because of the computer time necessary to run a full launch profile through the estimator, only the first stage of the truth model was used while testing at a one second data rate. The filter was able to detect the changing acceleration and corrected towards the true solution. However, the estimates started to lag, as the filter started to progress along the trajectory. The lag occurred in all the states but was especially noticeable in acceleration. By the end of the profile the filter was 2.3 g's less than that of the true acceleration. This is shown in Figure 7.

At first, it was thought there was an observability problem in the filter. The covariance matrix varied by twelve orders magnitude between the corresponding elements for position and acceleration. However, the eigenvalues of the covariance matrix were positive. This indicated that the matrix was positive definite. After further investigation, it was found that velocity and acceleration elements of the covariance had decreased by five and eight orders of magnitude respectively, during the run. So, as the covariance steadily decreased towards zero, the filter put more emphasis on the dynamics model and less on the data.

To correct this problem, a fading memory was added to the filter in which elements of the covariance matrix are deweighted to reflect decreased confidence in the seventh state. This was accomplished by multiplying the inverse covariance matrix, $\underline{P}^{-1}(t_i^-)$, by a scalar, β, just after propagation. β took on values between zero and one, depending upon the amount of fading desired. If a value of one was used, the filter retained its full expanding memory. No memory was retained for a zero value and the filter would revert back to a basic least squares estimator.

Incorporating the fading memory, the filter demonstrated improved performance. After testing many values of β, it was found that as the memory of the filter was decreased the estimation of the acceleration profile became better. In

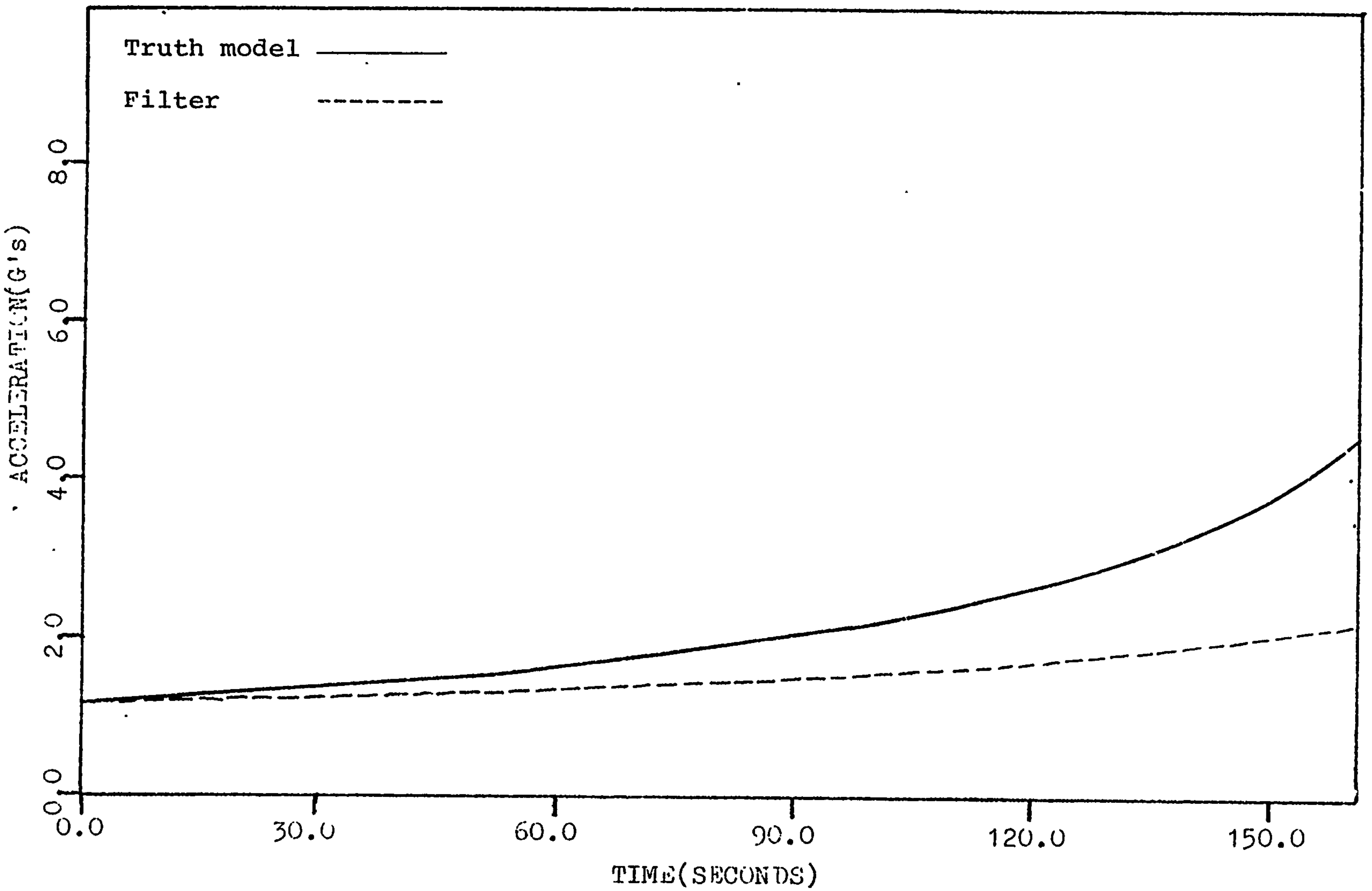


Figure 7. 7-state filter vs. truth model: 1 second perfect data

fact with β equal to zero, the filter gave a very good estimation of the acceleration profile as shown in Figure 8. However, the estimates of the position and velocity were on the high side. A β of 0.01 gave the filter its best performance using perfect data. Figure 9 shows the filter's estimate of acceleration using this β.

The filter had demonstrated the ability to follow an increasing variable acceleration profile. The full launch profile was given to the filter next, to determine its ability to follow a staging event. This time five measurements of perfect data at five second intervals were used for filter update. Again, sets of five were used for ease of implementation and to limit computer time. It was found that the filter had to have at least three measurements in order to make all seven states observable. On the other side, if large amounts of measurements were used, the filter's constant acceleration approximation would become less valid.

At first, it was thought that with the increased data interval, observability in the filter might be decreased. However the covariance matrix became better conditioned. Instead of the twelve orders of magnitude found with the one second data, it was now only nine. The increase in time between measurements made it easier for the filter to observe changes in the states.

With deviations, in the initial estimate of the first and seventh states, the filter demonstrated good performance in recognizing the staging events and correcting back to the

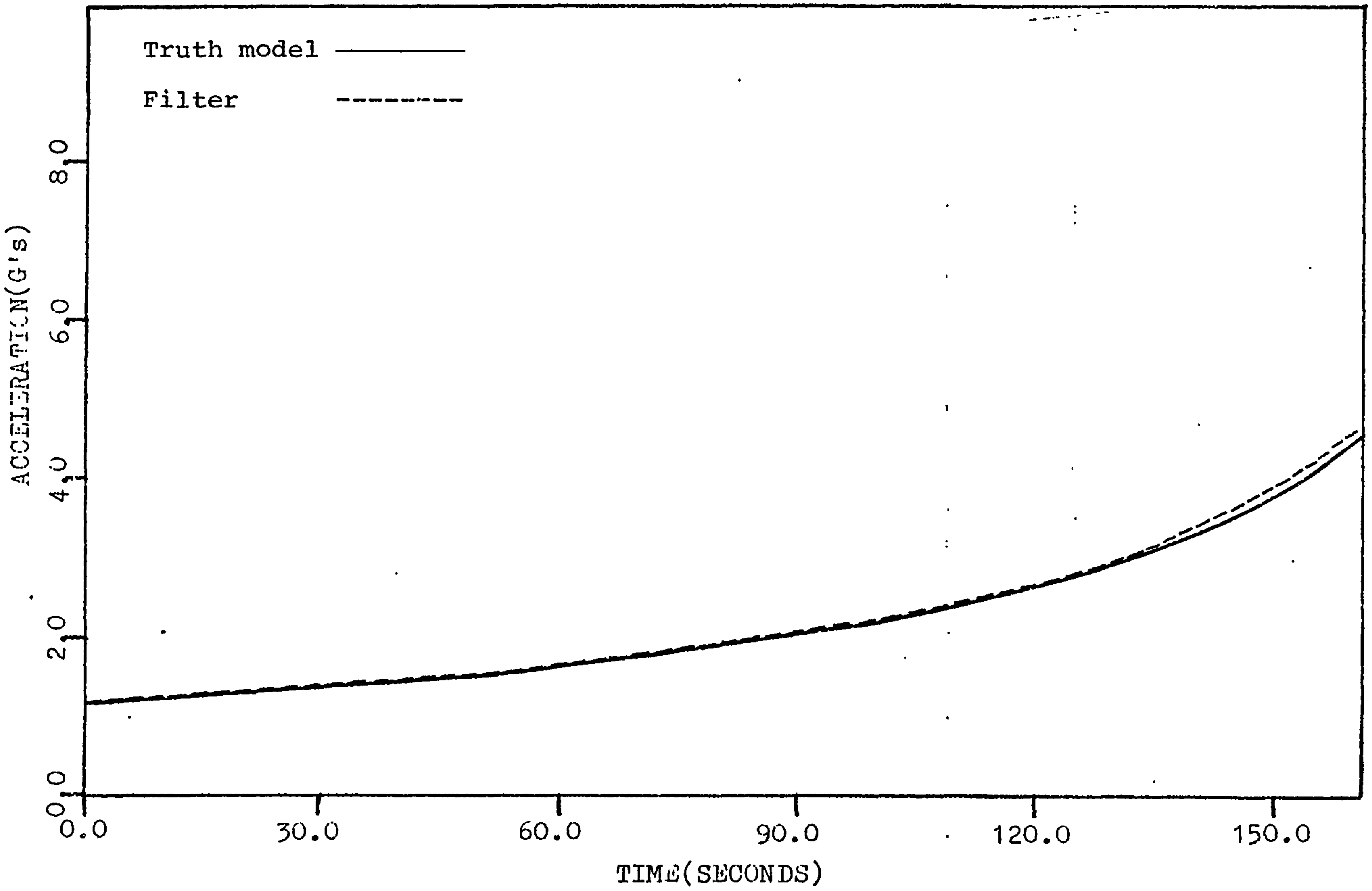


Figure 8. 7-state filter vs. truth model: 1 second perfect data, β = 0.0.

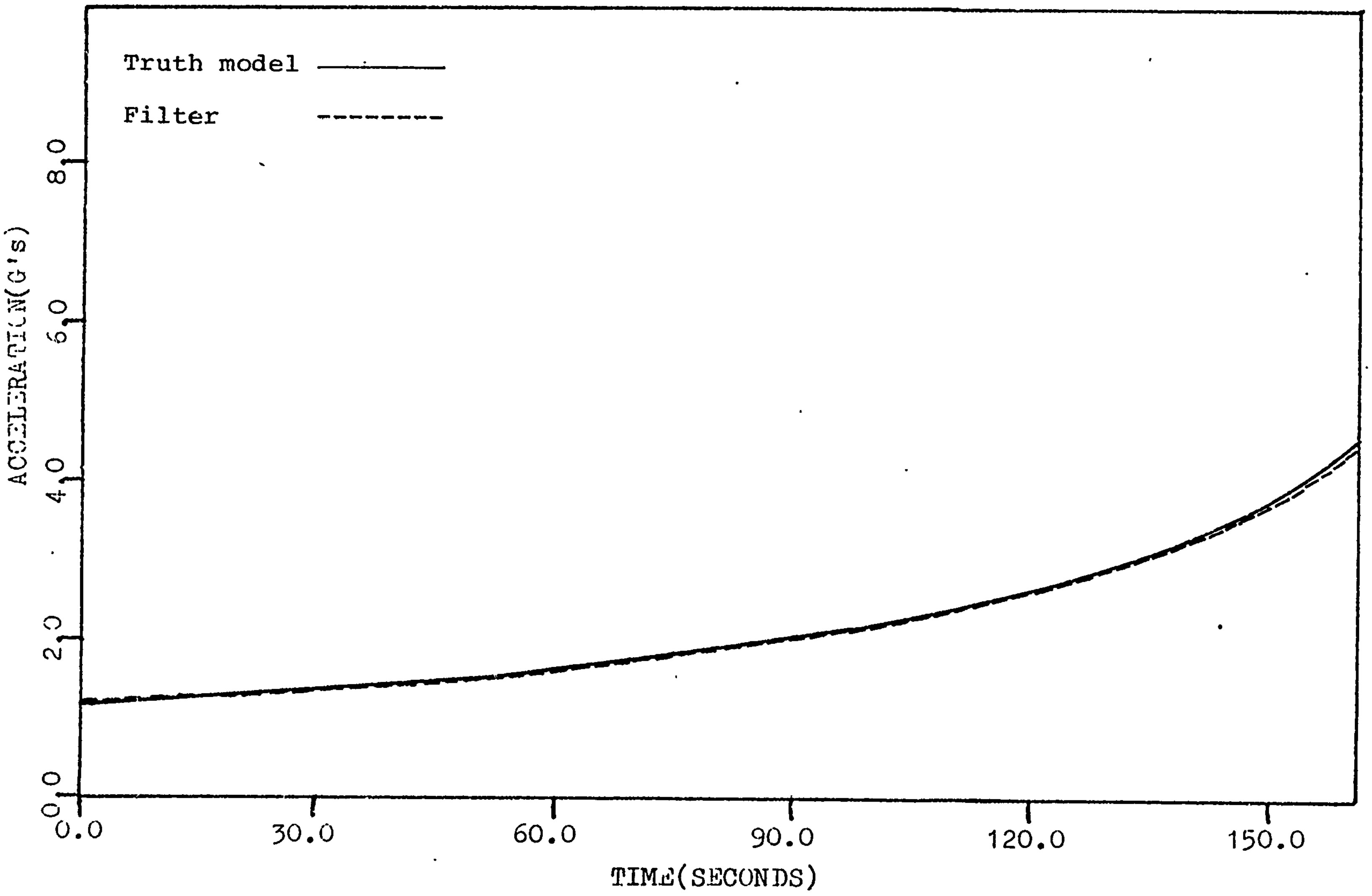


Figure 9. 7-state filter vs. truth model: 1 second perfect data, β = 0.01.

true solution. Again, with limited fading, the filter lagged in its estimates, which is evidenced in Figure 10. By experimentation, it was found that a β of 0.06 gave the best performance for the five second measurement interval (Figure 11). As the data span for an update was increased, less fading was required to obtain the filter's optimum performance. More emphasis had to be put on the dynamics model as the constant acceleration approximation became less valid.

An initial goal was to determine the filter's capabilities using a ten second data rate. With this measurement interval, three measurements were required for the update. As before, a minimum of three was needed t make all the states observable. However, with more than a thirty second data span, the change in acceleration became too great for the filter to handle, and excessive corrections to the states caused the filter to diverge. Figures 12, 13, and 14 show the filter's estimates of the vehicle's acceleration profile using various β's. A β of 0.1 gave the best estimation of the acceleration without significant degradation to the other state estimates. Various initial estimates were also tried in the filter. With poor estimates in all the states, the filter diverged quickly. With perfect estimates in the position states and degraded estimates in the other four states, the filter was able to converge to a solution. This showed favorable results, since a good initial guess in position of a rocket launch would be achievable while velocity and acceleration are less well known.

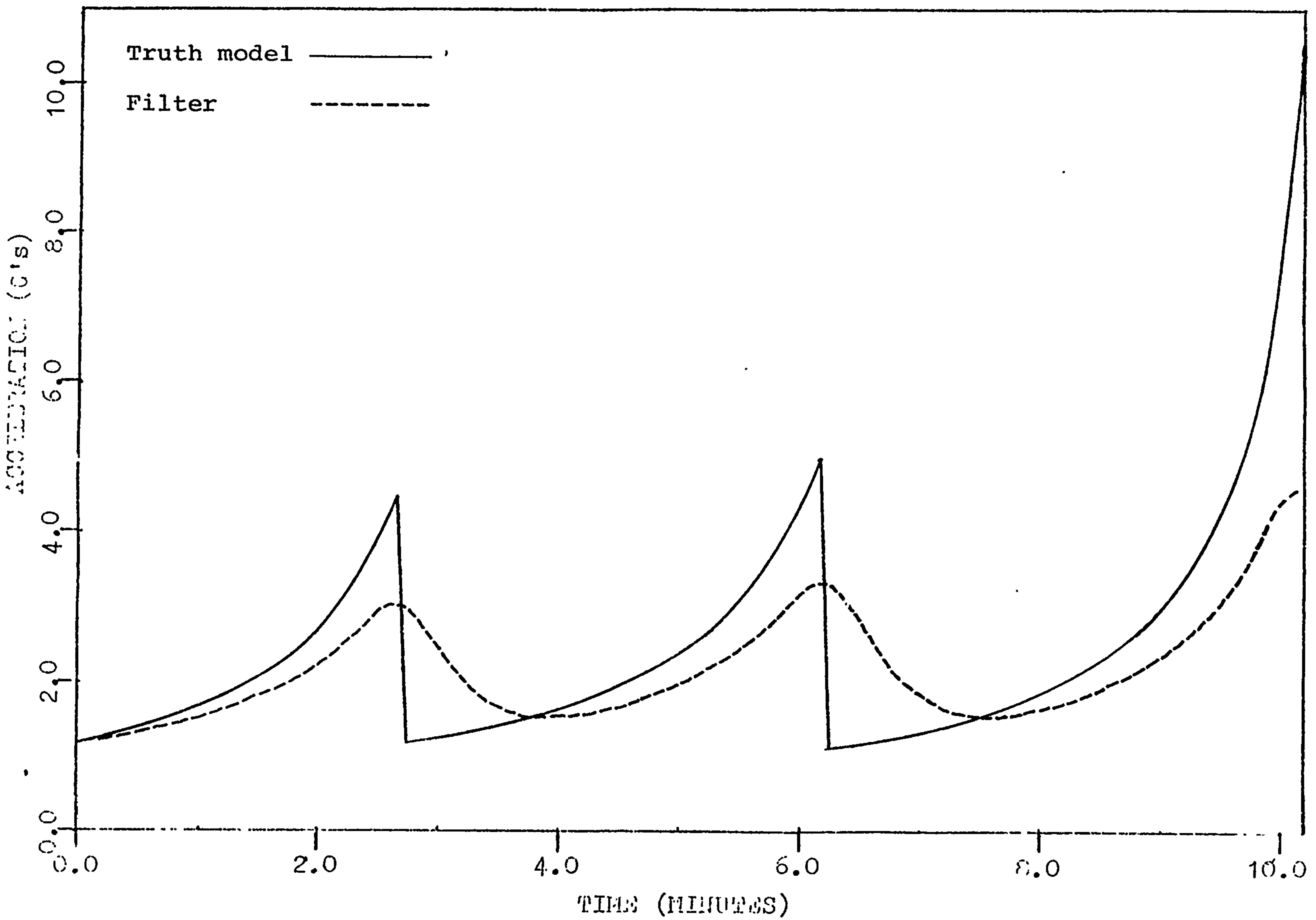


Figure 10. 7-state filter vs. truth model: 5 second perfect data, β = 0.5.

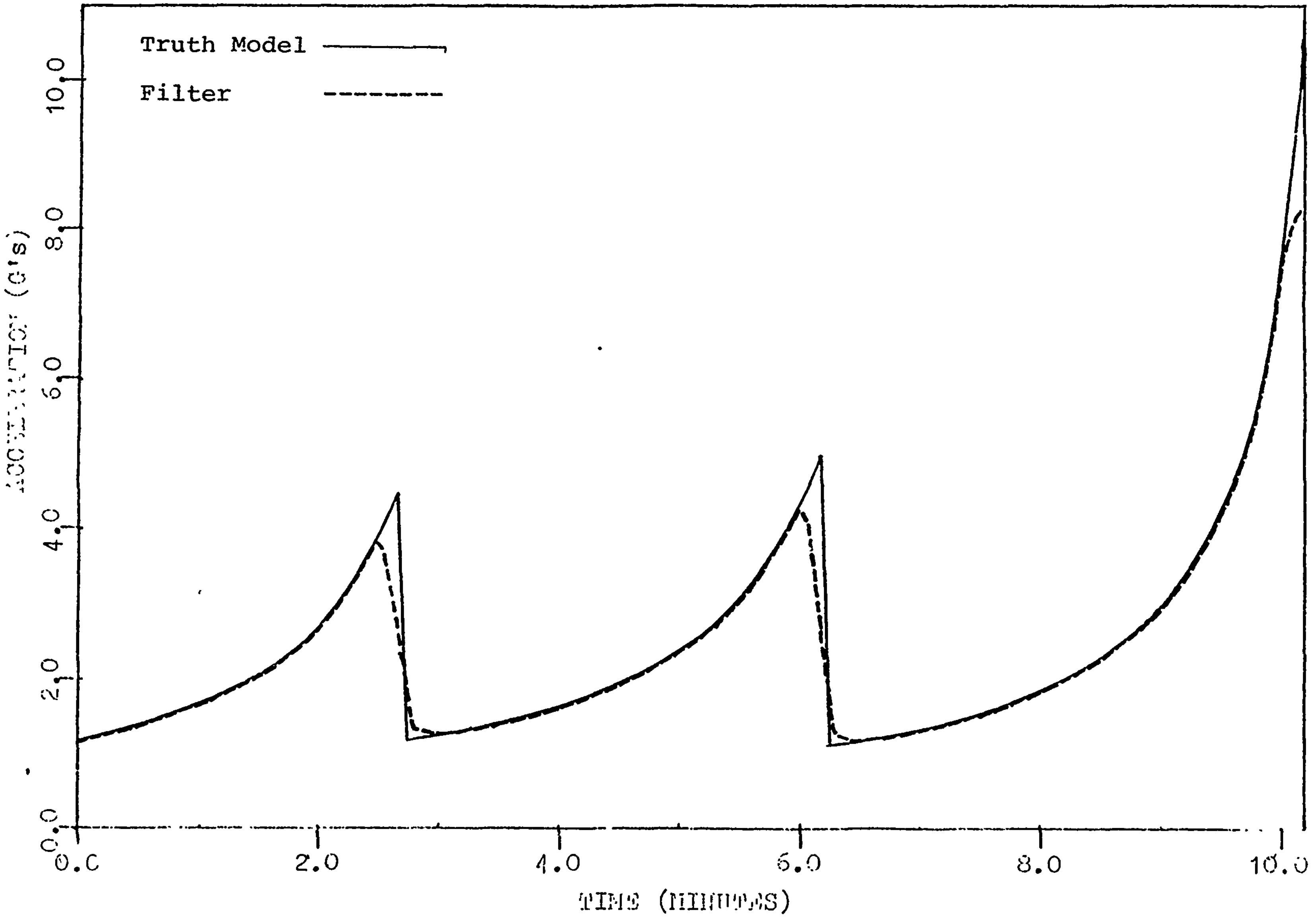


Figure 11. 7-state filter vs. truth model: 5 second perfect data, β = 0.06.

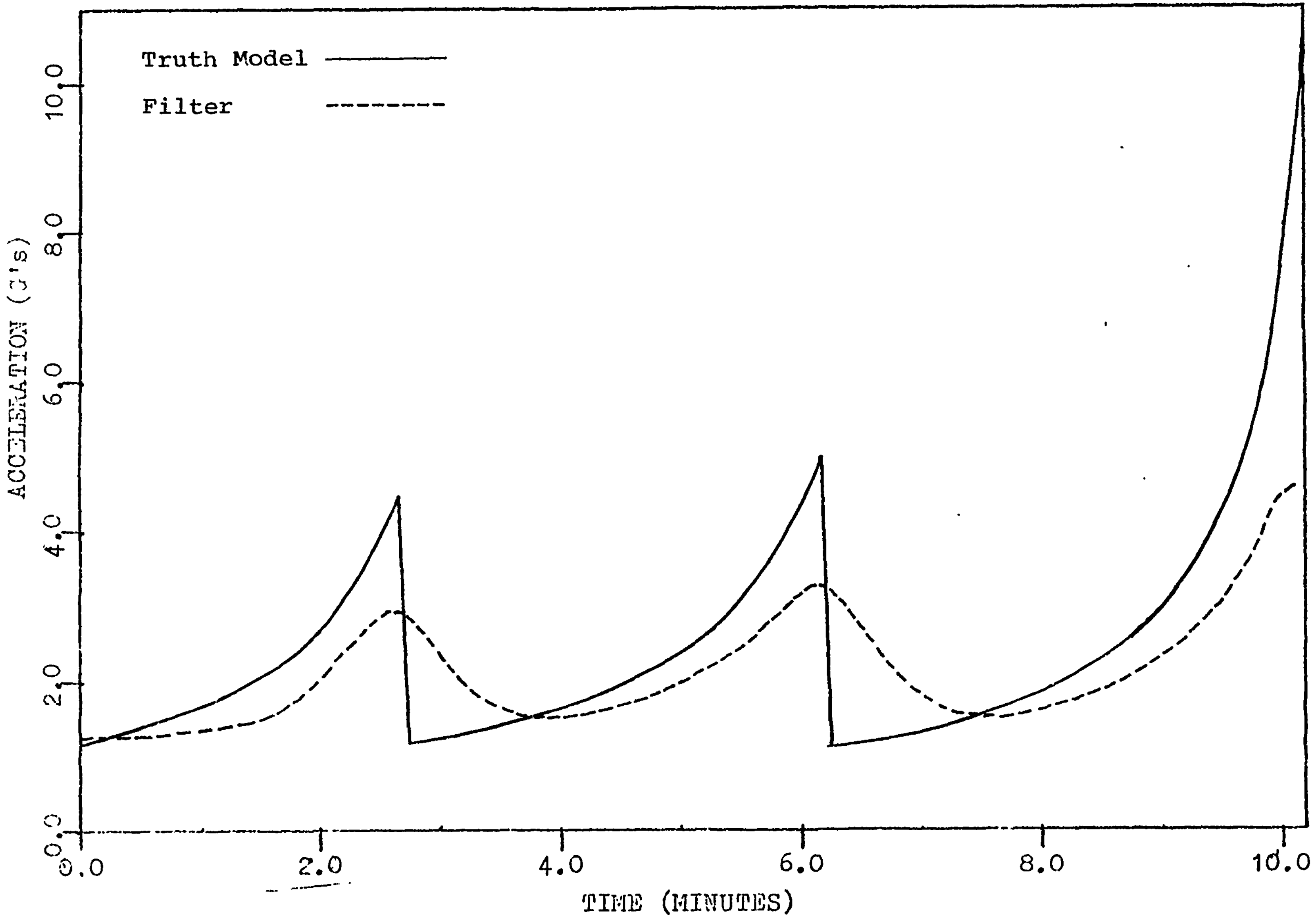


Figure 12. 7-state filter vs. truth model: 10 second perfect data, β = 0.5.

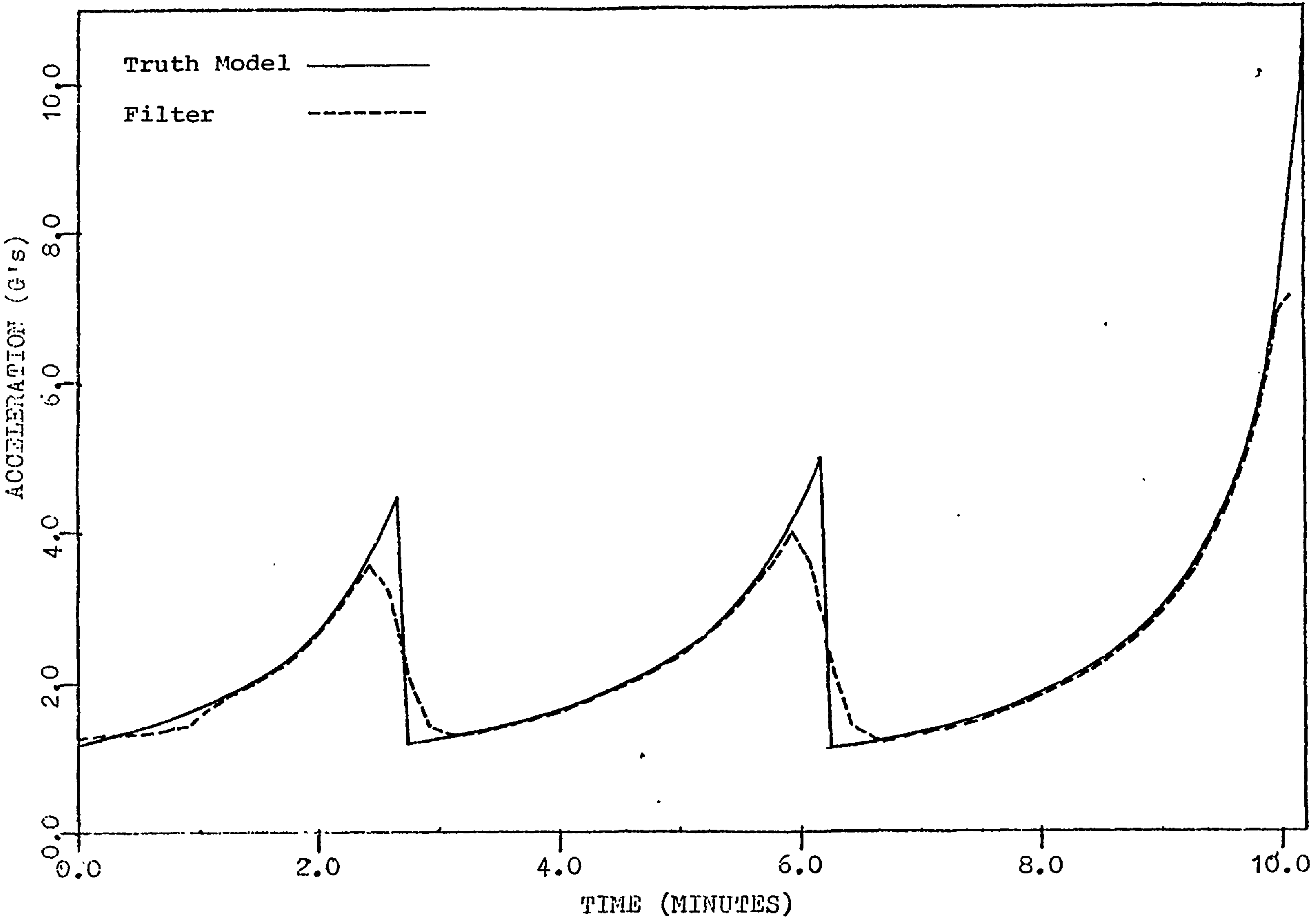


Figure 13. 7-state filter vs. truth model: 10 second perfect data, β = 0.1.

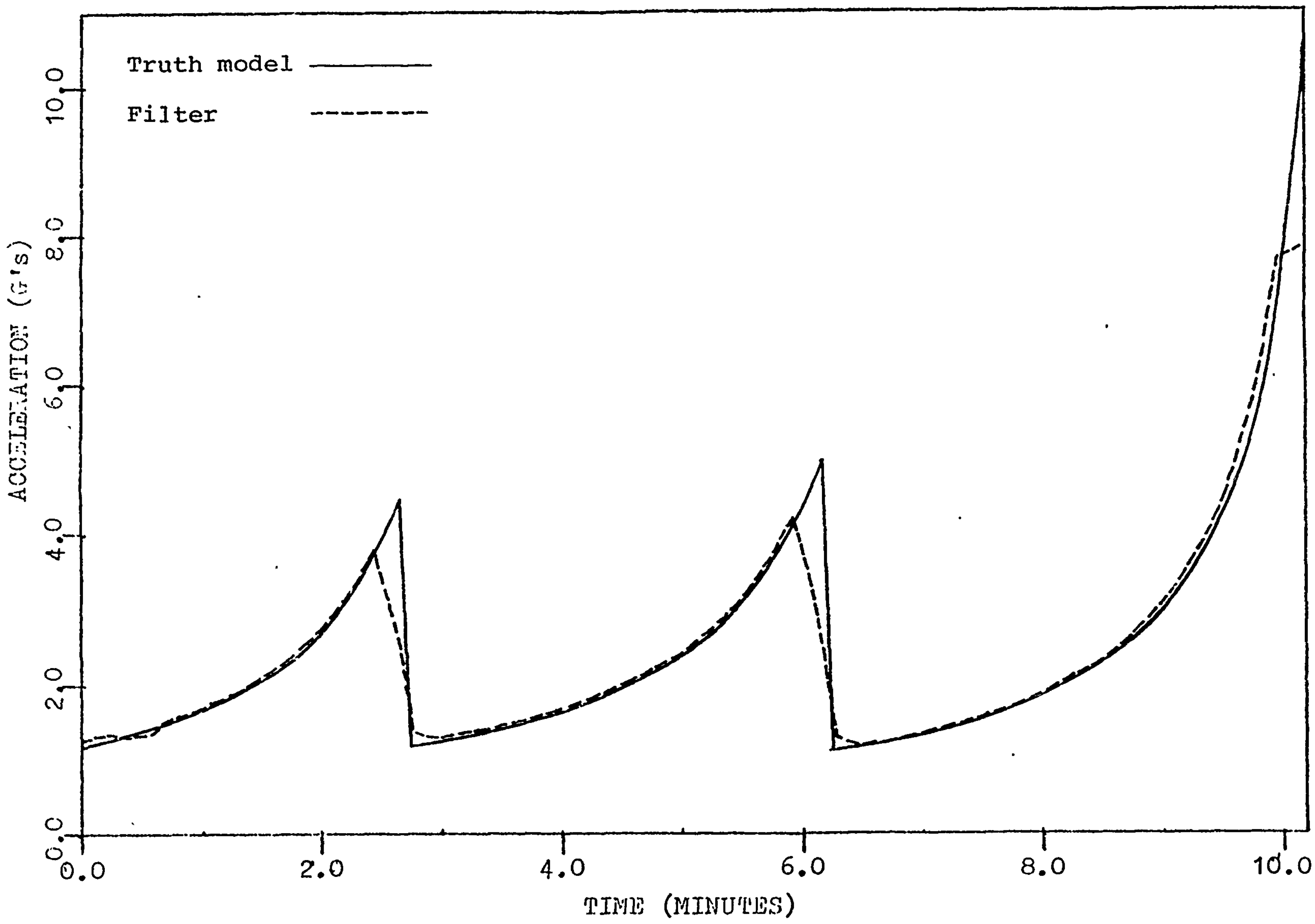


Figure 14. 7-state filter vs. truth model: 10 second perfect data, β = 0.01.

The filter had shown the ability to estimate the acceleration profile of a launch vehicle. However, this had been done with the use of perfect data. The measurements were assumed to be unbiased and uncorrelated in time. So zero-mean white Gaussian noise was added to the measurement angles to simulate imperfect data. The standard deviation (σ) of the noise was steadily increased to determine when the performance of the filter became degraded. There was no serious degradation in performance until a σ of 1.0×10^{-6} radians. This standard deviation equates to approximately 150 feet. Figure 15 shows the filter's estimate initially dropped off but then recovered. At launch with the vehicle moving slowly off the pad, the error ellipsoids start to overlap as more corruption is introduced into the measurements. The filter had trouble discerning changes in the states of the vehicle at liftoff. As the vehicle moved faster, the filter was better able to distinguish changes and correct back towards the true solution.

Increasing β and expanding the filter's memory produced increased performance as the noise was increased. With a β of 0.3, the filter was able to follow the vehicle's acceleration profile up to a σ of 1.0×10^{-5} (Figure 16). No improvement of performance was noted with further increases in β. The highest noise level attained before filter convergence could no longer be achieved was with $\sigma = 5.0 \times 10^{-5}$ (approximately 6800 feet). Although no information is given about the first stage (Figure 17), the second and third stage

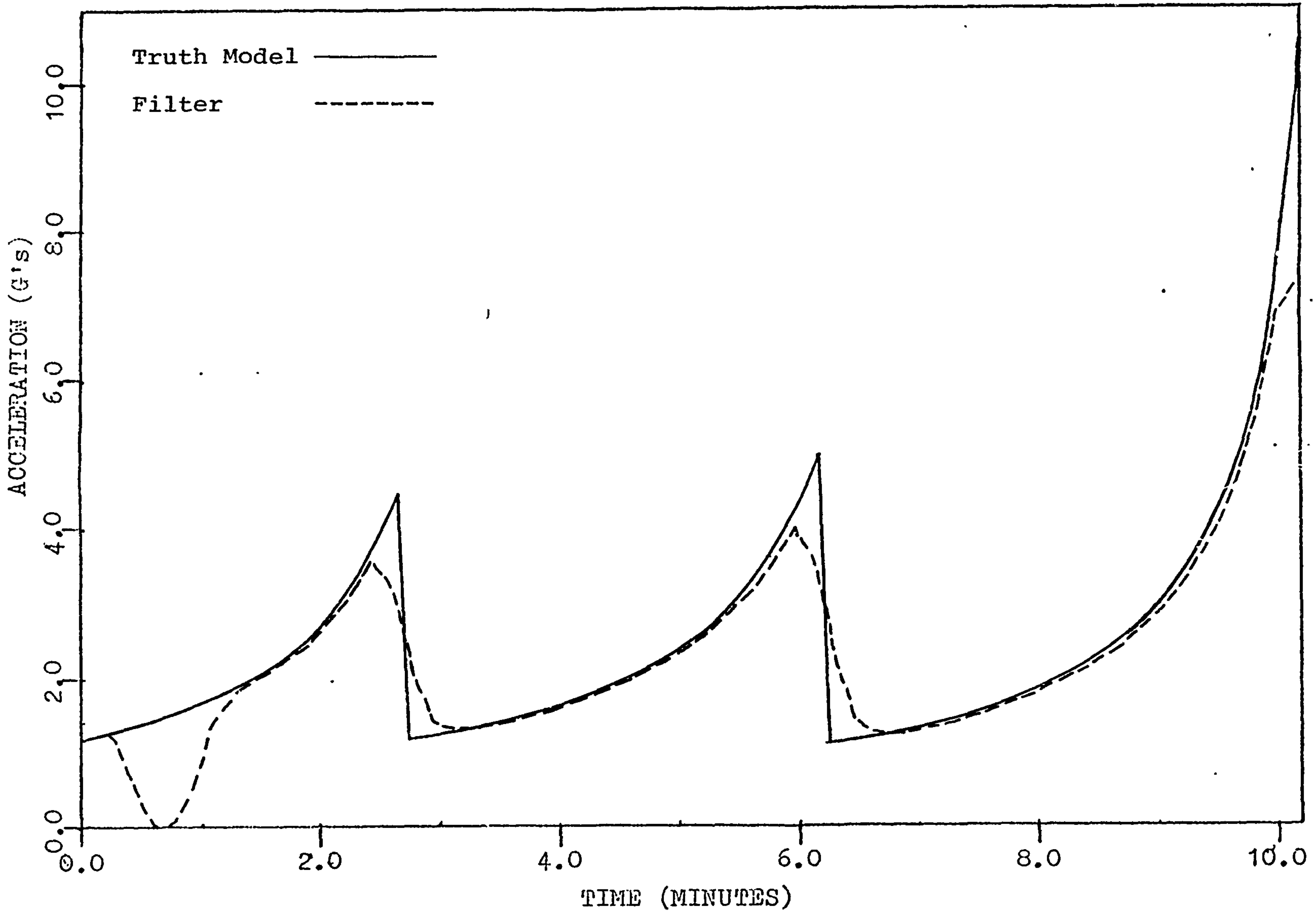


Figure 15. 7-state filter vs. truth model: 10 second data, $\sigma = 1.0 \times 10^{-5}$, $\beta = 0.3$.

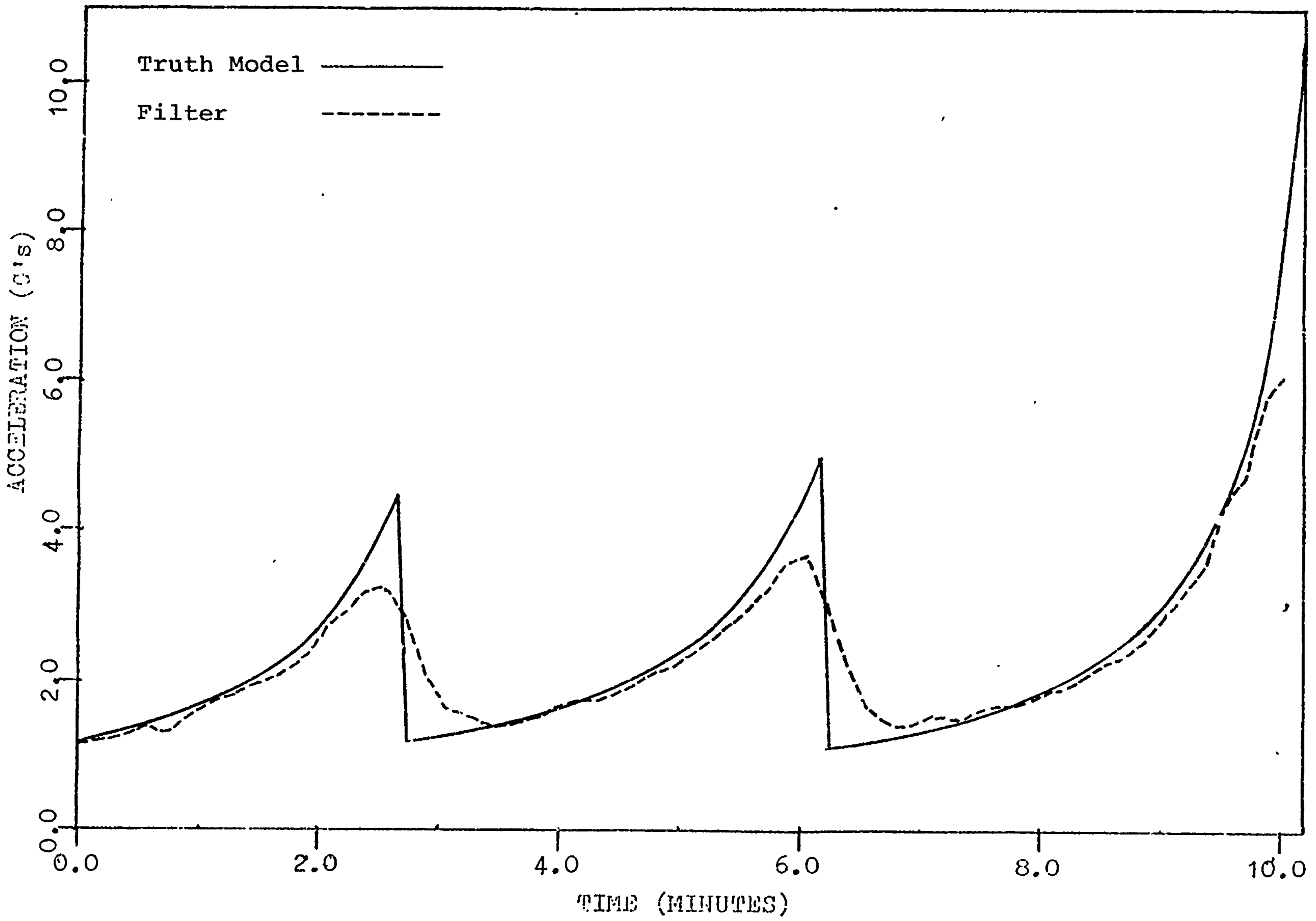


Figure 16. 7-state filter vs. truth model: 10 second data, $\sigma = 1.0 \times 10^{-5}$, $\beta = 0.3$.

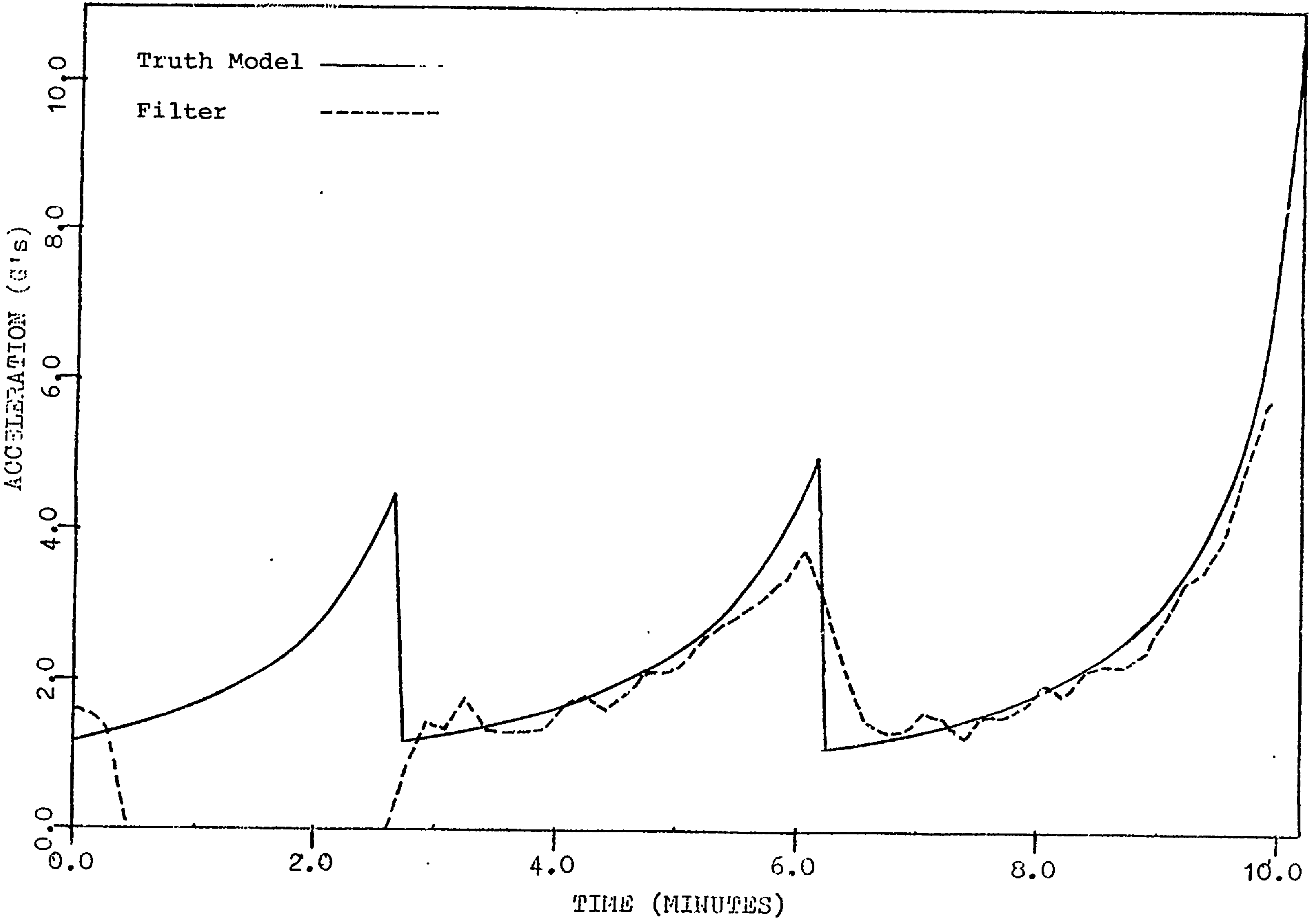


Figure 17. 7-state filter vs. truth model: 10 second data, $\sigma = 5.0 \times 10^{-5}$,

estimates give a good approximation of the actual acceleration profile. With an optimal smoother, these estimates could possibly be made better. However, this was not investigated.

The Eight State Filter

By making the appropriate changes in the $\overline{F}$ and $\overline{G}$ vectors and the $\underline{A}$ and $\underline{H}$ matrices, the eight state filter was easily implemented. The filter was initially given data with one second between observations. Having a better model for acceleration, it was thought that the eight state filter could handle a greater amount of data for update. Data for the entire first stage was batched to the filter for a least squares estimate. The filter diverged within three iterations. Twenty measurements were tried with initial estimate deviations in the seventh and eighth state. The filter was able to converge to the correct solution in two iterations. However, when the time interval between observations was increased to ten seconds, the covariance became ill-conditioned and the filter diverged within four iterations (Table 3). The eigenvalues of the covariance matrix had a spread of 16 orders of magnitude. Several other numbers of measurements were tried with no improvement in the filter's observability. With no significant results obtained, the attention of the study was directed towards the seven state filter.

Ideally, the performance parameters of a missile (M and V_e) would be obtained directly with the eight state filter. By running the seven state filter, the data for individual stages could be separated. Then the data could be run separately in the eight state filter to estimate the performance parameters of each stage. Estimates of position and velocity at staging obtained from the seven state filter could be used to start the eight state filter. With the performance parameter estimates of each stage, the acceleration profile could be computed and compared with the profile obtained from the seven state filter and get a better estimate However, the eight state model, as developed in the study, did not produce the desired results.

Table 3. Demonstrated filter instability (8-state)

Filter $\hat{\bar{x}}_{\text{initial}}$		Δx from true	Corrections			
			1st	2nd	3rd	4th
1	.1474736	0	.1089E-4	-.1777E-3	.8439E-2	-.1134E+13
2	-.8695592	-.001	.9290E-3	.8686E-4	.4222E-1	.6660E+13
3	.4731659	0	.4499E-4	-.4055E-4	.2317E-1	-.3636E+13
4	.0005689	0	-.1001E-2	.3010E-2	-.4045E+0	.8746E+14
5	-.0033487	0	.5865E-2	-.6365E-2	.2344E+1	-.5128E+15
6	.0018243	0	-.3203E-2	.3643E-2	-.1281E+1	.2801E+15
7	.318	.0001	.3148E+0	.7562E+1	-.1678E+4	.5034E+18
8	3.72	-.0002	-.2930E+1	-.1195E+2	.1969E+4	-.5950E+18

Notes: Correction based on 20 observations
Observations at 10 second intervals

VI Conclusions and Recommendations

In this study, an inverse covariance or Bayes filter was developed to estimate the performance parameters of a launch vehicle. Two orbital observers with assumed angles only (IR) measurements were used for filter update. A seven state and an eight state dynamics model were evaluated in the filter.

The seven state filter modelled acceleration as constant and lagged in its estimates of the launch profile. Addition of a fading memory to the filter improved performance significantly. Although a Monte Carlo analysis was not performed, due to initial problems encountered during the study, the filter showed good performance while varying initial conditions, data rates, trajectories and noise levels. It achieved a solution, estimating a launch profile with a ten second data rate and a noise level of $\sigma = 5.0 \times 10^{-5}$ radians.

Results indicate that variable acceleration cannot be estimated with an eight state filter with acceleration modelled using engine exit velocity, launch vehicle mass and propellant mass flow rate. The observability became worse with a higher dimensioned state vector.

It is recommended that further study be directed towards the seven state filter. Investigation into a fading memory differential corrector might be warranted. Using residual monitoring for adaptive choice of β might provide a better estimate during a staging event or just after lift-off. Also an optimal smoother might be considered to provide a better estimation profile using noisy data. Lastly, examination of using alternative measurements, such as range or range rate, is also recommended.

Because of failure of the eight state filter, direct estimation of the rocket engine performance parameters was not possible. Knowing that:

$$a(t) = \frac{a_o}{1 - M(t - t_o)} \qquad (2-22)$$

where $a_o = V_eM$ (initial acceleration)
a two state filter could be derived with a_o and M as the states. Acceleration estimates from seven state filter would be used as the data to estimate the two states for each stage. With estimates of a_o and M, V_e can be obtained directly.

Bibliography

1. Anthony, 1Lt John F., "Design of an Orbital Element Estimator using Relative Motion Data", Masters Thesis. Wright Patterson AFB, Ohio: Air Force Institute of Technology, December 1981.

2. Barker, Maj. Raymond H., "Estimation of Decayed Satellite Reentry Trajectories", Doctoral Dissertation. Wright Patterson AFB, Ohio: Air Force Institute of Technology, November 1981.

3. Bate, Roger R., Mueller, Donald D., and White, Jerry E. Fundamentals of Astrodynamics. New York: Dover Publications, Inc., 1971.

4. Maybeck, Peter S. Stochastic Models, Estimation, and Control. Vol. 1 New York: Academic Press, Inc., 1979.

5. Maybeck, Peter S. Stochastic Models, Estimation, and Control. Vol. 2 New York: Academic Press Inc., 1982.

6. Miller, Capt. Gregory D., "Estimation of Launch Vehicle Performance Parameters from an Orbiting Sensor", Masters Thesis. Wright Patterson AFB, Ohio: Air Force Institute of Technology, December 1981.

7. Morrison, Norman. Introduction to Sequential Smoothing and Prediction. New York, St. Louis, and San Francisco: McGraw-Hill Book Company, 1969.

8. U.S. Space Launch Systems (u), Navy Space Systems Activity. Report No. NSSA-R-20-72-2. P. O. Box 92960, Worldway Postal Center, Los Angeles, California, 90009.

9. Wiesel, William E., Jr. Lecture Material distributed in MC 731, Modern Methods of Orbit Determination. School of Engineering, Air Force Institute of Technology, Wright Patterson AFB, Ohio, 1981.

Appendix A

Derivation of the $\underline{A}$ Matrix

The elements of the $\underline{A}$ matrix are found by taking the gradient of the dynamics vector, $\bar{F}$.

$$\underline{A} = \nabla \bar{F} \tag{A-1}$$

where the elements are given by

$$A_{ij} = \frac{\partial F_i}{\partial x_j} \tag{A-2}$$

For the seven state filter:

$$\underline{A} = \begin{bmatrix} \frac{\partial \dot{x}_1}{\partial x_1} & \frac{\partial \dot{x}_1}{\partial x_2} & \frac{\partial \dot{x}_1}{\partial x_3} & \frac{\partial \dot{x}_1}{\partial x_4} & \frac{\partial \dot{x}_1}{\partial x_5} & \frac{\partial \dot{x}_1}{\partial x_6} & \frac{\partial \dot{x}_1}{\partial x_7} \\ \frac{\partial \dot{x}_2}{\partial x_1} & & & & & & \vdots \\ \frac{\partial \dot{x}_3}{\partial x_1} & & & & & & \vdots \\ \frac{\partial \dot{x}_4}{\partial x_1} & & & & & & \vdots \\ \frac{\partial \dot{x}_5}{\partial x_1} & & & & & & \vdots \\ \frac{\partial \dot{x}_6}{\partial x_1} & & & & & & \vdots \\ \frac{\partial \dot{x}_7}{\partial x_1} & \cdots & \cdots & \cdots & \cdots & \cdots & \frac{\partial \dot{x}_7}{\partial x_7} \end{bmatrix} \tag{A-3}$$

Using the state equations in Chapter II, the nonzero elements of $\underline{A}$ are:

$$A_{14} = \frac{\partial \dot{x}}{\partial v_x} = 1 \tag{A-4}$$

$$A_{25} = \frac{\partial \dot{y}}{\partial v_y} = 1 \tag{A-5}$$

$$A_{36} = \frac{\partial \dot{z}}{\partial v_z} = 1 \tag{A-6}$$

$$A_{41} = \frac{\partial \dot{v}_x}{\partial x} = -\frac{\mu}{r^3} + \frac{3\mu x^2}{r^5} \tag{A-7}$$

$$A_{42} = \frac{\partial \dot{v}_x}{\partial y} = \frac{3\mu xy}{r^5} \tag{A-8}$$

$$A_{43} = \frac{\partial \dot{v}_x}{\partial z} = \frac{3\mu xz}{r^5} \tag{A-9}$$

$$A_{44} = \frac{\partial \dot{v}_x}{\partial v_x} = -\frac{v_x^2 a}{v^3} + \frac{a}{v} \tag{A-10}$$

$$A_{45} = \frac{\partial \dot{v}_x}{\partial v_y} = -\frac{v_x v_y a}{v^3} \tag{A-11}$$

$$A_{46} = \frac{\partial \dot{v}_x}{\partial v_z} = -\frac{v_x v_z a}{v^3} \tag{A-12}$$

$$A_{47} = \frac{\partial \dot{v}_x}{\partial a} = \frac{v_x}{v} \tag{A-13}$$

$$A_{51} = \frac{\partial \dot{v}_y}{\partial x} = \frac{3\mu yx}{r^3} \tag{A-14}$$

$$A_{52} = \frac{\partial \dot{v}_y}{\partial y} = -\frac{\mu}{r^3} + \frac{3\mu y^2}{r^5} \tag{A-15}$$

$$A_{53} = \frac{\partial \dot{v}_y}{\partial z} = \frac{3\mu yz}{r^5} \tag{A-16}$$

$$A_{54} = \frac{\partial \dot{v}_y}{\partial v_x} = -\frac{v_y v_x a}{v^3} \tag{A-17}$$

$$A_{55} = \frac{\partial \dot{v}_y}{\partial v_y} = -\frac{v_y^2 a}{v^3} + \frac{a}{v} \tag{A-18}$$

$$A_{56} = \frac{\partial \dot{v}_y}{\partial v_z} = -\frac{v_y v_z a}{v^3} \tag{A-19}$$

$$A_{57} = \frac{\partial \dot{v}_y}{\partial a} = \frac{v_y}{v} \tag{A-20}$$

$$A_{61} = \partial \dot{v}_{-} = \frac{3\mu xz}{r^5} \tag{A-21}$$

$$A_{62} = \frac{\partial \dot{v}_z}{\partial y} = \frac{3\mu yz}{r^5} \tag{A-22}$$

$$A_{63} = \frac{\partial \dot{v}_z}{\partial z} = -\frac{\mu}{r^3} + \frac{3\mu z^2}{r^5} \tag{A-23}$$

$$A_{64} = \frac{\partial \dot{v}_z}{\partial v_x} = -\frac{v_x v_z a}{v^3} \tag{A-24}$$

$$A_{65} = \frac{\partial \dot{v}_z}{\partial v_y} = -\frac{v_y v_z a}{v^3} \tag{A-25}$$

$$A_{66} = \frac{\partial \dot{v}_z}{\partial v_z} = -\frac{v_z^2 a}{v^3} + \frac{a}{v} \tag{A-26}$$

$$A_{67} = \frac{\partial \dot{v}_z}{\partial a} = \frac{v_z}{v} \qquad \text{(A-27)}$$

where

$$r = \sqrt{x^2+y^2+z^2}$$

$$v = \sqrt{v_x^2+v_y^2+v_z^2}$$

and other variables are given in Chapter II.

For the eight state filter, the A matrix is 8 x 8 with six new non-zero terms:

$$A_{47} = \frac{v_x a}{vV_e} \qquad \text{(A-28)}$$

$$A_{48} = \frac{v_x}{v}\left(\frac{a^2 t}{V_e M} + \frac{a}{M}\right) \qquad \text{(A-29)}$$

$$A_{57} = \frac{v_y a}{vV_e} \qquad \text{(A-30)}$$

$$A_{58} = \frac{v_y}{v}\left(\frac{a^2 t}{V_e M} + \frac{a}{M}\right) \qquad \text{(A-31)}$$

$$A_{67} = \frac{v_z a}{vV_e} \qquad \text{(A-32)}$$

$$A_{68} = \frac{v_z}{v}\left(\frac{a^2 t}{V_e M} + \frac{a}{M}\right) \qquad \text{(A-33)}$$

Appendix B

Derivation of $\underline{H}$ Matrix

The elements of the $\underline{H}$ matrix are found by taking the partial derivatives of elevation and azimuth with respect to all elements of the state vector. For the seven state filter, $\underline{H}$ is a 2 by 7 matrix given by:

$$\underline{H} = \frac{\partial \underline{G}}{\partial \overline{x}} = \begin{bmatrix} \frac{\partial el}{\partial x} & \frac{\partial el}{\partial y} & \frac{\partial el}{\partial z} & \frac{\partial el}{\partial v_x} & \frac{\partial el}{\partial v_y} & \frac{\partial el}{\partial v_z} & \frac{\partial el}{\partial a} \\ \frac{\partial az}{\partial x} & \frac{\partial az}{\partial y} & \frac{\partial az}{\partial z} & \frac{\partial az}{\partial v_x} & \frac{\partial az}{\partial v_y} & \frac{\partial az}{\partial v_z} & \frac{\partial az}{\partial a} \end{bmatrix} \qquad \text{(B-1)}$$

Since $\underline{G}$ is only a function of the three position elements of the state vector, the last four columns of the matrix are zero.

Partial derivatives of elevation:

$$el = \sin^{-1} \left[\frac{\left| - \overline{R} \frac{(\overline{r} \cdot \overline{R})}{|\overline{R}|\,|\overline{R}|} + \overline{r} \right|}{|\overline{r}-\overline{R}|} \right] \qquad \text{(B-2)}$$

$$\frac{d}{dx} (\sin^{-1} u) = \frac{1}{\sqrt{1-u^2}} \frac{du}{dx} \qquad \text{(B-3)}$$

$$|\overline{r}| = \sqrt{x^2+y^2+z^2} \qquad \text{(B-4)}$$

where x,y,z are position components for the target vehicle

$$|\overline{R}| = \sqrt{R_x^2+R_y^2+R_z^2} \qquad \text{(B-5)}$$

where R_x, R_y, R_z are position components of the observer.

Using eqs (B-2) and (B-3):

$$\frac{\partial el}{\partial x} = \left[1 - \left[\frac{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\;|\bar{R}|} + \bar{r}\right|}{|\bar{r} - \bar{R}|}\right]^2\right]^{-\frac{1}{2}} \frac{d}{dx}\left[\frac{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\;|\bar{R}|} + \bar{r}\right|}{|\bar{r} - \bar{R}|}\right] \qquad \text{(B-6)}$$

$$\frac{\partial el}{\partial x} = \left[1 - \left[\frac{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\;|\bar{R}|} + \bar{r}\right|}{|\bar{r} - \bar{R}|}\right]^2\right]^{-\frac{1}{2}} \left[(-1)\left[\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\;|\bar{R}|} + \bar{r}\right|\right](|\bar{r} - \bar{R}|)^{-2} + \left|\frac{-\bar{R}(R_x)}{|\bar{R}|\,|\bar{R}|} + 1\right|\left[|\bar{r} - \bar{R}|\right]^{-1}\right] \qquad \text{(B-7)}$$

Similarly:

$$\frac{\partial el}{\partial y} = \frac{\left[\frac{(-1)\left[\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|\right]}{(|\bar{r} - \bar{R}|)^2} + \frac{\left|\frac{-\bar{R}(R_y)}{|\bar{R}|\,|\bar{R}|} + 1\right|}{|\bar{r} - \bar{R}|}\right]}{\sqrt{1 - \left[\frac{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|}{|\bar{r} - \bar{R}|}\right]^2}} \qquad \text{(B-8)}$$

$$\frac{\partial el}{\partial z} = \frac{\left[\frac{(-1)\left[\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|\right]}{(|\bar{r} - \bar{R}|)^2} + \frac{\left|\frac{-R(R_z)}{|\bar{R}|\,|\bar{R}|} + 1\right|}{|\bar{r} - \bar{R}|}\right]}{\sqrt{1 - \left[\frac{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|}{|\bar{r} - \bar{R}|}\right]^2}} \qquad \text{(B-9)}$$

Partial derivatives of azimuth:

$$az = \cos^{-1}\left[\frac{\left[\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right]\cdot\hat{k}}{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|}\right] \qquad \text{(B-10)}$$

$$\frac{d}{dx}\left(\cos^{-1}u\right) = \frac{-1}{\sqrt{1-u^2}}\frac{du}{dx} \tag{B-11}$$

From eqs (B-10) and (B-11)

$$u = \left[\frac{\left[\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|}\right] + \bar{r} \quad \cdot \hat{k}}{\left|\frac{-\bar{R}(\bar{r}\cdot\bar{R})}{|\bar{R}|\,|\bar{R}|} + \bar{r}\right|}\right] \tag{B-12}$$

the numerator is:

$$\frac{-R_z(xR_x+yR_y+zR_z)}{R_x^2+R_y^2+R_z^2} + z$$

Let: $\text{DOT} = xR_x+yR_y+zR_z$

$\text{RSQD} = R_x^2+R_y^2+R_z^2$

So the numerator can be written: $z - \frac{R_z\ \text{DOT}}{\text{RSQD}}$

$$u^2 = \frac{\left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2}{\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RQSD}}\right)^2 + \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2} \tag{B-13}$$

$$\frac{du}{dx} = \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right) \cdot \quad (-\tfrac{1}{2}) \cdot \left[\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right)^2\right.$$

$$\left. + \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2\right]^{-3/2} \cdot \left[(2) \cdot \left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right) \cdot \left(1 - \frac{R_x^2}{\text{RSQD}}\right)\right.$$

$$\left. + (2) \cdot \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right) \cdot \left(\frac{-R_xR_y}{\text{RSQD}}\right) + (2) \cdot \left(z - \frac{R_x\text{DOT}}{\text{RSQD}}\right) \cdot \left(\frac{-R_xR_z}{\text{RSQD}}\right)\right]$$

$$+ \frac{\left(\frac{-R_xR_z}{\text{RSQD}}\right)}{\sqrt{\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right)^2 + \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2}} \qquad \text{(B-14)}$$

so $$\frac{d}{dx}(\cos^{-1}u) = \frac{-1}{\sqrt{1 - \frac{\left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2}{\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right)^2 + \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2}}}$$

$$\cdot \left[\frac{(-1) \quad \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right) \cdot \left[\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right) \cdot \left(1 - \frac{R_x^2}{\text{RSQD}}\right) + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right) \cdot \left(\frac{-R_xR_y}{\text{RSQD}}\right) + \left(z - \frac{R_x\text{DOT}}{\text{RSQD}}\right) \cdot \left(\frac{-R_x}{\text{RSQ}}\right.\right.}{\left[\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right)^2 + \left(z - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2\right]^{3/2}}\right.$$

$$\left. + \frac{\left(\frac{-R_xR_z}{\text{RSQD}}\right)}{\sqrt{\left(x - \frac{R_x\text{DOT}}{\text{RSQD}}\right)^2 + \left(y - \frac{R_y\text{DOT}}{\text{RSQD}}\right)^2 + \left(z - \frac{R_z\text{DOT}}{\text{RSQD}}\right)^2}}\right] \qquad \text{(B-15)}$$

Similarly:

$$\frac{d}{dy}\,(\cos^{-1}u) = \frac{-1}{\sqrt{1-\dfrac{\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2}{\cdot\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2}}}$$

$$\cdot\left[\frac{(-1)\cdot\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left[\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{PSQD}}\right)\cdot\left(\dfrac{-R_xR_y}{\mathrm{RSQD}}\right)+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left(1-\dfrac{R_y^2}{\mathrm{RSQD}}\right)+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left(\dfrac{-R_y}{\mathrm{RSQ}}\right.\right.}{\left[\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2\right]^{3/2}}\right.$$

$$\left.+\frac{\left(\dfrac{-R_yR_z}{\mathrm{RSQD}}\right)}{\sqrt{\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2}}\right] \qquad \text{(B-16)}$$

and:

$$\frac{d}{dz}\,(\cos^{-1}u) = \frac{-1}{\sqrt{1-\dfrac{\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2}{\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(z-\dfrac{R_z\mathrm{LOT}}{\mathrm{RSQD}}\right)^2}}}$$

$$\cdot\left[\frac{(-1)\;\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left[\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left(\dfrac{-R_xR_z}{\mathrm{RSQD}}\right)+\left(y-\dfrac{R_y\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left(\dfrac{-R_yR_z}{\mathrm{RSQD}}\right)+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)\cdot\left(1-\dfrac{}{\mathrm{R}}\right.\right.}{\cdot\left[\left(x-\dfrac{R_x\mathrm{DOT}}{\mathrm{RSQD}}\right)^2+\left(y-\dfrac{R_y\mathrm{LOT}}{\mathrm{RSQD}}\right)^2+\left(z-\dfrac{R_z\mathrm{DOT}}{\mathrm{RSQD}}\right)^2\right]^{3/2}}\right.$$

$$+ \frac{\left(1 - \frac{R_z^2}{RSQD}\right)}{\sqrt{\left(x-\frac{R_x DOT}{RSQD}\right)^2 + \left(y-\frac{R_y DOT}{RSQD}\right)^2 + \left(z-\frac{R_z DOT}{RSQD}\right)^2}} \qquad \text{(B-17)}$$

The sign of eqs B-15, B-16, and B-17 was determined by looking at the z component of the cross product $\bar{R} \times \bar{r}$. If the z component was negative the sign was negative.

For the eight state filter $\underline{H}$ became a 2 x 8 matrix with the added elements being zero.

Appendix C
Bayes Filter (Seven State) Program

```
      PROGRAM BAYES(INPUT,OUTPUT,TAPE7)
      EXTERNAL F
      REAL  XREF(7),DELTA,PHI(7,7),Y(56),TIME,ELMAT,AZMAT,ELMAT2,
     +   AZMAT2,T,TOUT,RELERR,ABSERR,WORK(1276),IWORK(5),GMAT(2,1),
     +   GMAT2(2,1),XOBS(3),XOBS2(3),HMAT(4,7),HMAT1(2,7),HMAT2(2,7)
     +   H(4,7),HTRANS(7,4),Q(4,4),QINV(4,4),P1(4,7),PINV(7,7),D1,D2
     +   DELX(7,1),B1(4,1),B2(7,1),WKAREA,QV,QZ,PV,PZ,TMAT(7,7),
     +   XNEW(7),HTQR(7,1),R(4,1),XMNS(7),XDIFF(7,1),PMNSI(7,7),
     +   STR(7,1),PX(7,1),PPLS(7,7),PP(7,7),PMNS(7,7),PHIT(7,7),
     +   HTQH(7,7),BETA(7,7),PMNSIO(7,7),XPLOT(122),YPLOT(122),
     +   EA,EP,EV,TT(611),X(611),YY(611),Z(611),VX(611),VY(611),
     +   VZ(611),A(611),EX2,EY2,EZ2,EVX2,EVY2,EVZ2,EP2,EV2,EA2,
     +   RMSP,RMSV,RMSA,WORK2(7),PSIV(7,7),ZZ(7)
      INTEGER  COUNT,NUM,I,J,IDGT,IER,TI,CODE,ITER,ITER2
      DIMENSION  QV(10),QZ(10),PV(28),PZ(28),WKAREA(250),
     +   TIME(0:160),ELMAT(160),AZMAT(160),ELMAT2(160),AZMAT2(160)
      PARAMETER(   COUNT=61)
*     ...INITIALIZE...
      XREF(1)=.71009116
      XREF(2)=-.41827566
      XREF(3)=.56640624
      XREF(4)=100*COS(-.53249995)*COS(0.60213859)/25936.24764
      XREF(5)=100*SIN(-.53249995)*COS(0.60213859)/25936.24764
      XREF(6)=100*SIN(0.60213859)/25936.24764
      XREF(7)=1.184
      TI=0
      TIME(0)=0.0
      ITER=1
      ITER2=3
      K=0
      ICODE=1
      M=1
      EP=0.0
      EV=0.0
      EA=0.0
      EP2=0.0
      EV2=0.0
      EA2=0.0
      MEAN=0.0
      STD=0.0
      CALL RANSET(77)
      DO 2 I=1,7
         DO 2 J=1,7
         XMNS(I)=XREF(I)
            PMNSI(I,J)=0.0
         BETA(I,J)=0.0
 2    CONTINUE
      BETA(1,1)=0.1
      BETA(2,2)=0.1
      BETA(3,3)=0.1
      BETA(4,4)=0.1
      BETA(5,5)=0.1
      BETA(6,6)=0.1
      BETA(7,7)=0.1
```

```
        DO 5 I=1,56
           Y(I)=0.0
5       CONTINUE
        DO 10 I=1,4
           DO 10 J=1,4
              Q(I,J)=0.0
10      CONTINUE
        DO 15 I=1,4
           Q(I,I)=1.0E-14
15      CONTINUE
*       ...FIND Q INVERSE...
        CALL CHNGV(Q,4,QV)
        CALL LINV2P(QV,4,QZ,IDGT,D1,D2,WKAREA,IER)
        CALL CHNGM(QZ,4,QINV)
*       ...READ IN DATA AND ADD NOISE...
        DO 20 NUM=1,COUNT
        READ (5,*)TIME(NUM),ELMAT(NUM),AZMAT(NUM),ELMAT2(NUM),AZMAT2(N
        ELMAT(NUM)=ELMAT(NUM)+GAUSS(MEAN,STD)
        AZMAT(NUM)=AZMAT(NUM)+GAUSS(MEAN,STD)
        ELMAT2(NUM)=ELMAT2(NUM)+GAUSS(MEAN,STD)
        AZMAT2(NUM)=AZMAT2(NUM)+GAUSS(MEAN,STD)
20      CONTINUE
        DO 200 I=1,610
           READ(4,*)X(I),YY(I),Z(I),VX(I),VY(I),VZ(I),A(I)
200     CONTINUE
        PRINT*,' TIME=0.0 '
*       ...COMPUTE RESIDUALS AND H MATRICES...
22      DO 25 I=1,7
           DO 25 J=1,7
              HTQH(I,J)=0.0
              HTQR(I,1)=0.0
              PHI(I,J)=0.0
              XNEW(I)=XREF(I)
25      CONTINUE
        DO 27 I=1,7
           PHI(I,I)=1.0
27      CONTINUE
        DO 50 NUM=ITER,ITER2
        T=TIME(NUM-1)
           TOUT=TIME(NUM)
           RELERR=1.0E-07
           ABSERR=1.0E-07
           NEQN=56
           IFLAG=-1
           DO 30 I=1,7
        DO 30 J=1,7
              Y(I)=XNEW(I)
              Y(I*7+J)=PHI(I,J)
30         CONTINUE
        CALL ODE(F,NEQN,Y,T,TOUT,RELERR,ABSERR,IFLAG,WORK,IWORK)
        DO 35 I=1,7
           DO 35 J=1,7
              PHI(I,J)=Y(I*7+J)
              XNEW(I)=Y(I)
```

```
35     CONTINUE
*      ...FIND POSITION OF OBSERVERS AND CALCULATE G MATRIX AND
*         RESIDUALS...
       CALL ELVAZ(TIME(NUM),XNEW,GMAT,GMAT2,XOBS,XOBS2)
       R(1,1)=ELMAT(NUM)-GMAT(1,1)
       R(2,1)=AZMAT(NUM)-GMAT(2,1)
       R(3,1)=ELMAT2(NUM)-GMAT2(1,1)
       R(4,1)=AZMAT2(NUM)-GMAT2(2,1)
*       ...FIND H MATRICES...
       CALL MATH(XNEW(1),XNEW(2),XNEW(3),XOBS(1),XOBS(2),XOBS(3),HMAT
       CALL MATH(XNEW(1),XNEW(2),XNEW(3),XOBS2(1),XOBS2(2),XOBS2(3),
     +    HMAT2)
       DO 40 J=1,7
          HMAT(1,J)=HMAT1(1,J)
          HMAT(2,J)=HMAT1(2,J)
          HMAT(3,J)=HMAT2(1,J)
          HMAT(4,J)=HMAT2(2,J)
40     CONTINUE
*      ...MULTIPLY H BY PHI...
       CALL MMPY(HMAT,4,7,PHI,7,H)
*      ...FIND H TRANSPOSE...
       DO 45 I=1,4
          DO 45 J=1,7
             HTRANS(J,I)=H(I,J)
45     CONTINUE
*      ...FIND HTRANS*Q INVERSE*H...
       CALL MMPY(QINV,4,4,H,7,P1)
       CALL MMPY(HTRANS,7,4,P1,7,TMAT)
       DO 55 I=1,7
          DO 55 J=1,7
             HTQH(I,J)=HTQH(I,J)+TMAT(I,J)
55     CONTINUE
*      ...FIND H TRANSPOSE*Q INVERSE*R
       CALL MMPY(QINV,4,4,R,1,B1)
       CALL MMPY(HTRANS,7,4,B1,1,B2)
       DO 60 I=1,7
          HTQR(I,1)=HTQR(I,1)+B2(I,1)
60     CONTINUE
50     CONTINUE
       DO 65 I=1,56
          Y(I)=0.0
65     CONTINUE
       K=K+1
*      ...FIND P PLUS...
       DO 70 I=1,7
          DO 70 J=1,7
             PINV(I,J)=PMNSI(I,J)+HTQH(I,J)
70     CONTINUE
       CALL CHNGV(PINV,7,PV)
       CALL LINV2P(PV,7,PZ,IDGT,D1,D2,WKAREA,IER)
       CALL CHNGM(PZ,7,PPLS)
*      ...COMPUTE EIGENVALUES OF UPDATED COVARIANCE...
       IF (K.EQ.1) THEN
       DO 72 I=1,7
```

```
            DO 72 J=1,7
               PSIV(I,J)=PPLS(I,J)
 72      CONTINUE
         CALL EIGRS(PSIV,7,10,ZZ,PV,7,WORK2,IER)
         PRINT*,' EIGENVALUES '
         DO 400 I=1,7
            PRINT*,'         ',ZZ(I)
 400     CONTINUE
         END IF
 *       ...COMPUTE UPDATE TO STATE VECTOR...
         DO 75 I=1,7
            XDIFF(I,1)=XMNS(I)-XREF(I)
 75      CONTINUE
         CALL MMPY(PMNSI,7,7,XDIFF,1,STR)
         DO 80 I=1,7
            PX(I,1)=STR(I,1)+HTQR(I,1)
 80      CONTINUE
         CALL MMPY(PPLS,7,7,PX,1,DELX)
         DO 85 I=1,7
            XREF(I)=XREF(I)+DELX(I,1)
 85      CONTINUE
         IF (K.EQ.1) THEN
         PRINT*,'P(+)= '
         PRINT'(7(7(1X,E15.8)/))',((PPLS(I,J),J=1,7),I=1,7)
         END IF
         PRINT'(7(1X,E15.8))',(DELX(I,1),I=1,7)
         PRINT*,'XREF= '
         PRINT'(7(1X,E15.8)/)',(XREF(I),I=1,7)
         IF (K.EQ.3) GO TO 86
 *       ...CHECK FOR CONVERGENCE...
         IF (ABS(DELX(1,1)).GT.PPLS(1,1)) GO TO 22
         IF (ABS(DELX(2,1)).GT.PPLS(2,2)) GO TO 22
         IF (ABS(DELX(3,1)).GT.PPLS(3,3)) GO TO 22
         IF (ABS(DELX(4,1)).GT.PPLS(4,4)) GO TO 22
         IF (ABS(DELX(5,1)).GT.PPLS(5,5)) GO TO 22
         IF (ABS(DELX(6,1)).GT.PPLS(6,6)) GO TO 22
         IF (ABS(DELX(7,1)).GT.PPLS(7,7)) GO TO 22
 86      XPLOT(M)=(TIME(TI)+0.0030986)*806.3113744
         YPLOT(M)=XREF(7)
         WRITE (7,110)XPLOT(M),YPLOT(M)
 110     FORMAT(F8.4,2X,E15.8)
         EX2=(XREF(1)-X(ICODE))**2
 *       ...COMPUTE DEVIATIONS IN STATES FROM TRUTH MODEL...
         EY2=(XREF(2)-YY(ICODE))**2
         EZ2=(XREF(3)-Z(ICODE))**2
         EVX2=(XREF(4)-VX(ICODE))**2
         EVY2=(XREF(5)-VY(ICODE))**2
         EVZ2=(XREF(6)-VZ(ICODE))**2
         EA=(XREF(7)-A(ICODE))**2
         EP=EX2+EY2+EZ2
         EV=EVX2+EVY2+EVZ2
         EP2=EP2+EP
         EV2=EV2+EV
         EA2=EA2+EA
```

```
*        ...PROPAGATE STATE AND COVARIANCE...
        T=TIME(TI)
        TI=TI+1
        TOUT=TIME(TI)
        RELERR=1.0E-07
        ABSERR=1.0E-07
        NEQN=56
        IFLAG=-1
           DO 87 I=1,7
              Y(I)=XREF(I)
              Y(I*8)=1.0
87         CONTINUE
        CALL ODE(F,NEQN,Y,T,TOUT,RELERR,ABSERR,IFLAG,WORK,IWORK)
        DO 90 I=1,7
           DO 90 J=1,7
              XREF(I)=Y(I)
              XNEW(I)=Y(I)
              XMNS(I)=Y(I)
              PHI(I,J)=Y(I*7+J)
              PHIT(J,I)=Y(I*7+J)
90      CONTINUE
*       ...FIND P MINUS AND P MINUS INVERSE...
        CALL MMPY(PPLS,7,7,PHIT,7,PP)
        CALL MMPY(PHI,7,7,PP,7,PMNS)
        CALL CHNGV(PMNS,7,PV)
        CALL LINV2F(PV,7,PZ,IDGT,D1,D2,WKAREA,IER)
        CALL CHNGM(PZ,7,PMNSIO)
*       ...MULTIPLY P MINUS INVERSE BY BETA...
        CALL MMPY(PMNSIO,7,7,BETA,7,PMNSI)
        PRINT'('' TIME= '',F7.5)',TIME(TI)
        PRINT*,'X MINUS= '
        PRINT'(7(1X,E15.8))',(XNEW(I),I=1,7)
        ITER=ITER+1
        IF (ITER.GE.COUNT) THEN
           GO TO 100
        END IF
        ITER2=ITER2+1
        IF (ITER2.GT.COUNT) THEN
           ITER2=COUNT
        END IF
        ICODE=ICODE+10
        K=0
        M=M+1
        DO 95 I=1,56
           Y(I)=0.0
95      CONTINUE
        GO TO 22
*       ...COMPUTE RMS ERRORS OF FILTER...
100     RMSP=SQRT(EP2/(M-1))
        RMSV=SQRT(EV2/(M-1))
        RMSA=SQRT(EA2/(M-1))
        PRINT'('' RMS ERROR POSITION= '',E15.8)',RMSP
        PRINT'('' RMS ERROR VELOCITY= '',E15.8)',RMSV
        PRINT'('' RMS ERROR ACCELERATION= '',E15.8)',RMSA
        END
```

```
      SUBROUTINE MMPY(A,IA,JA,B,JB,C)
*     ...MATRIX MULTIPLICATION ROUTINE...
      REAL A(IA,JA),B(JA,JB),C(IA,JB)
      DOUBLE PRECISION  TD
      DO 20 K=1,IA
         DO 20 J=1,JB
            TD=0.D00
            DO 10 I=1,JA
10             TD=TD+A(K,I)*B(I,J)
20    C(K,J)=TD
      END


      SUBROUTINE F(T,Y,YP)
*     ...SETS UP STATES AND PHI MATRIX FOR INTEGRATION...
      REAL Y(56),YP(56),PHI(7,7),FMAT(7,1),AMAT(7,7),PHIDOT(7,7),T
      CALL MATF(Y(1),Y(2),Y(3),Y(4),Y(5),Y(6),Y(7),FMAT)
      DO 10 I=1,7
         YP(I)=FMAT(I,1)
10    CONTINUE
      CALL MATA(Y(1),Y(2),Y(3),Y(4),Y(5),Y(6),Y(7),AMAT)
      DO 20 I=1,7
         DO 20 J=1,7
            PHI(I,J)=Y(I*7+J)
20    CONTINUE
      CALL MMPY(AMAT,7,7,PHI,7,PHIDOT)
      DO 30 I=1,7
         DO 30 J=1,7
            YP(I*7+J)=PHIDOT(I,J)
30    CONTINUE
      END


      SUBROUTINE MATF(X,Y,Z,VX,VY,VZ,A,FMAT)
*     ...COMPUTES F VECTOR
      REAL FMAT(7,1),X,Y,Z,VX,VY,VZ,A,VE,M,T,R3
      R3=(X**2+Y**2+Z**2)*SQRT(X**2+Y**2+Z**2)
      FMAT(1,1)=VX
      FMAT(2,1)=VY
      FMAT(3,1)=VZ
      FMAT(4,1)=-X/R3+A*VX/SQRT(VX**2+VY**2+VZ**2)
      FMAT(5,1)=-Y/R3+A*VY/SQRT(VX**2+VY**2+VZ**2)
      FMAT(6,1)=-Z/R3+A*VZ/SQRT(VX**2+VY**2+VZ**2)
      FMAT(7,1)=0.0
      END
      SUBROUTINE MATA(X,Y,Z,VX,VY,VZ,A,AMAT)
*     ...COMPUTES A MATRIX...
      REAL X,Y,Z,VX,VY,VZ,A,R3,R5,V3,AMAT(7,7),V1
      INTEGER I,J
      R3=(X**2+Y**2+Z**2)*SQRT(X**2+Y**2+Z**2)
      R5=((X**2+Y**2+Z**2)**2)*SQRT(X**2+Y**2+Z**2)
      V1=SQRT(VX**2+VY**2+VZ**2)
      V3=(VX**2+VY**2+VZ**2)*SQRT(VX**2+VY**2+VZ**2)
      DO 10 I=1,7
         DO 20 J=1,7
            AMAT(I,J)=0.0
20       CONTINUE
10    CONTINUE
      AMAT(1,4)=1
      AMAT(2,5)=1
      AMAT(3,6)=1
      AMAT(4,1)= -1/R3+3*X**2/R5
      AMAT(4,2)= 3*X*Y/R5
      AMAT(4,3)= 3*X*Z/R5
      AMAT(4,4)= A/V1-VX**2*A/V3
      AMAT(4,5)= -VX*VY*A/V3
      AMAT(4,6)= -VX*VZ*A/V3
      AMAT(4,7)= VX/V1
      AMAT(5,1)= 3*X*Y/R5
      AMAT(5,2)= -1/R3+3*Y**2/R5
      AMAT(5,3)= 3*Y*Z/R5
      AMAT(5,4)= -VX*VY*A/V3
      AMAT(5,5)= A/V1-VY**2*A/V3
      AMAT(5,6)= -VY*VZ*A/V3
      AMAT(5,7)= VY/V1
      AMAT(6,1)= 3*X*Z/R5
      AMAT(6,2)= 3*Y*Z/R5
      AMAT(6,3)= -1/R3+3*Z**2/R5
      AMAT(6,4)= -VX*VZ*A/V3
      AMAT(6,5)= -VY*VZ*A/V3
      AMAT(6,6)= A/V1-VZ**2*A/V3
      AMAT(6,7)= VZ/V1
      END


      SUBROUTINE MATG(X,Y,Z,RSUBX,RSUBY,RSUBZ,GMAT)
*     ...COMPUTES G MATRIX...
      REAL X,Y,Z,RSUBX,RSUBY,RSUBZ,EL,AZ,DIFF,DOT,RSQD,NUM,NUM1,
     +   CROSS,GMAT(2,1)
      RSQD=RSUBX**2+RSUBY**2+RSUBZ**2
      CROSS=RSUBX*Y-RSUBY*X
      DOT=X*RSUBX+Y*RSUBY+Z*RSUBZ
      NUM=SQRT((X-RSUBX*DOT/RSQD)**2+(Y-RSUBY*DOT/RSQD)**2+
     +      (Z-RSUBZ*DOT/RSQD)**2)
```

```
      DIFF=SQRT((X-RSUBX)**2+(Y-RSUBY)**2+(Z-RSUBZ)**2)
      GMAT(1,1)=ASIN(NUM/DIFF)
      NUM1=Z-RSUBZ*DOT/RSQD
      GMAT(2,1)=ACOS(NUM1/NUM)
      IF (CROSS .LT. 0.0) THEN
      GMAT(2,1)=-GMAT(2,1)
      END IF
      END


      SUBROUTINE MATH(X,Y,Z,RSUBX,RSUBY,RSUBZ,HMAT)
*     ...COMPUTES OBSERVATION RELATION MATRIX H...
      REAL HMAT(2,7),X,Y,Z,RSUBX,RSUBY,RSUBZ,RSQD,NUM,DIFF,DOT
      INTEGER I,J
      DO 10 I=1,2
         DO 20 J=4,7
            HMAT(I,J)=0.0
20       CONTINUE
10    CONTINUE
      RSQD=RSUBX**2+RSUBY**2+RSUBZ**2
      DOT=X*RSUBX+Y*RSUBY+Z*RSUBZ
      NUM=SQRT((X-RSUBX*DOT/RSQD)**2+(Y-RSUBY*DOT/RSQD)**2+(Z-RSUBZ
     +   *DOT/RSQD)**2)
      DIFF=SQRT((X-RSUBX)**2+(Y-RSUBY)**2+(Z-RSUBZ)**2)
      CROSS=RSUBX*Y-RSUBY*X
      HMAT(1,1)=((NUM)*(RSUBX-X)/DIFF**3+(X*(RSUBY**2+RSUBZ**2)
     +   -RSUBX*DOT*(RSUBY**2+RSUBZ**2)/RSQD-Y*RSUBX*RSUBY+DOT*RSUBX
     +   *RSUBY**2/RSQD-Z*RSUBX*RSUBZ+DOT*RSUBX*RSUBZ**2/RSQD)/DIFF
     +   /NUM/RSQD)/SQRT(1-(NUM/DIFF)**2)
      HMAT(1,2)=((NUM)*(RSUBY-Y)/DIFF**3+(DOT*RSUBX**2*RSUBY/RSQD
     +   -X*RSUBX*RSUBY-DOT*RSUBY*(RSUBX**2+RSUBZ**2)/RSQD
     +   +Y*(RSUBX**2+RSUBZ**2)+DOT*RSUBY*RSUBZ**2/RSQD
     +   -Z*RSUBY*RSUBZ)/DIFF/NUM/RSQD)/SQRT(1-(NUM/DIFF)**2)
      HMAT(1,3)=((NUM)*(RSUBZ-Z)/DIFF**3+(DOT*RSUBX**2*RSUBZ/RSQD
     +   -X*RSUBX*RSUBZ+DOT*RSUBY**2*RSUBZ/RSQD-Y*RSUBY*RSUBZ
     +   -DOT*RSUBZ*(RSUBX**2+RSUBY**2)/RSQD+Z*(RSUBX**2+RSUBY**2))
     +   /DIFF/NUM/RSQD)/SQRT(1-(NUM/DIFF)**2)
      HMAT(2,1)=(-1)/SQRT(1-((Z-RSUBZ*DOT/RSQD)/NUM)**2)
     +   *(((-RSUBX*RSUBZ/RSQD)/NUM)-((Z-RSUBZ*DOT/RSQD)/NUM**3
     +   *((X-RSUBX*DOT/RSQD)*((RSUBY**2+RSUBZ**2)/RSQD)-(Y-RSUBY*DOT
     +   /RSQD)*(RSUBX*RSUBY/RSQD)-(Z-RSUBZ*DOT/RSQD)
     +   *(RSUBX*RSUBZ/RSQD))))
      HMAT(2,2)=(-1)/SQRT(1-((Z-RSUBZ*DOT/RSQD)/NUM)**2)
     +   *(((-RSUBY*RSUBZ/RSQD)/NUM)-((Z-RSUBZ*DOT/RSQD)/NUM**3
     +   *((X-RSUBX*DOT/RSQD)*(-RSUBX*RSUBY/RSQD)+(Y-RSUBY*DOT/RSQD)*
     +   ((RSUBX**2+RSUBZ**2)/RSQD)-(Z-RSUBZ*DOT/RSQD)
     +   *(RSUBY*RSUBZ/RSQD))))
```

```
      HMAT(2,3)=(-1)/SQRT(1-((Z-RSUBZ*DOT/RSQD)/NUM)**2)
     +  *(((1-RSUBZ**2/RSQD)/NUM)-((Z-RSUBZ*DOT/RSQD)/NUM**3
     +   *((X-RSUBX*DOT/RSQD)*(-RSUBX*RSUBZ/RSQD)-(Y-RSUBY*DOT/RSQD)
     +   *(RSUBY*RSUBZ/RSQD)+(Z-RSUBZ*DOT/RSQD)*((RSUBX**2+RSUBY**2)
     +   /RSQD))))
      IF (CROSS .LT. 0.0) THEN
         HMAT(2,1)=-HMAT(2,1)
         HMAT(2,2)=-HMAT(2,2)
         HMAT(2,3)=-HMAT(2,3)
      END IF
      END


      SUBROUTINE ELNAZ(TIME,XR,GMAT,GMAT2,XOBS,XOBS2)
*     ...CALCULATES OBSERVERS' POSITION FOR G MATRIX COMPUTATION...
      REAL  TIME,XR(7),GMAT(2,1),GMAT2(2,1),XOBS(3),XOBS2(3)
      XOBS(1)=6.6*COS((2*3.141592654*TIME)/106.53585)
      XOBS(2)=6.6*SIN((2*3.141592654*TIME)/106.53585)
      XOBS(3)=0.0
      XOBS2(1)=6.6*SIN((2*3.141592654*TIME)/106.53585)
      XOBS2(2)=-6.6*COS((2*3.141592654*TIME)/106.53585)
      XOBS2(3)=0.0
      CALL MATG(XR(1),XR(2),XR(3),XOBS(1),XOBS(2),XOBS(3),GMAT)
      CALL MATG(XR(1),XR(2),XR(3),XOBS2(1),XOBS2(2),XOBS2(3),GMAT2)
      END


      SUBROUTINE CHNGV(AMAT,N,AVECT)
*     ...CHANGES MATRIX INTO VECTOR FORM FOR INVERSE ROUTINE...
      REAL AMAT(N,N),AVECT(N*(N+1)/2)
      INTEGER N,J,I,ICNT
      ICNT=0
      DO 10 I=1,N
         DO 10 J=1,I
            ICNT=ICNT+1
               AVECT(ICNT)=AMAT(I,J)
10    CONTINUE
      END
      SUBROUTINE CHNGM(A,N,NMAT)
*     ...CHANGES VECTOR BACK INTO MATRIX AFTER INVERSION...
      REAL A(N*(N+1)/2),NMAT(N,N)
      INTEGER N,ICNT,I,J
      ICNT=0
      DO 10 I=1,N
         ICNT=ICNT+I
         DO 10 J=1,I
            NMAT(I,J)=A(ICNT+J-I)
            NMAT(J,I)=A(ICNT+J-I)
10    CONTINUE
      END


      REAL FUNCTION GAUSS(MEAN,STD)
*     ...RANDOM NOISE GENERATOR...
      REAL  MEAN
      SUM=-6.
      DO 1 I=1,12
1     SUM=SUM+RANF()
      GAUSS=STD*SUM+MEAN
      END
```

VITA

Donald W. Gross [redacted] [redacted]

PII Redacted [redacted] [redacted] graduated from high school in Attleboro, Massachusetts in 1969. He attended the United States Air Force Academy and received a Bachelor of Science Degree in Astronautical Engineering in 1973. He completed pilot training and received his wings in August 1974. He served as a F-4E aircraft commander with the 22nd Tactical Fighter Squadron at Bitburg Air Base, Germany and the 512th Tactical Fighter Squadron at Ramstein Air Base, Germany. Then he served as a T-38A/B instructor pilot in the Lead-In Fighter Training program with the 479 Tactical Training Wing at Holloman Air Force Base, New Mexico, until entering the School of Engineering, Air Force Institute of Technology, in June 1981.

[redacted] [redacted]

[redacted]

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
AFIT/GA/AA/82D-4

2. GOVT ACCESSION NO.
AD A124 802

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
ESTIMATION OF LAUNCH VEHICLE PERFORMANCE PARAMETERS FROM TWO ORBITING SENSORS

5. TYPE OF REPORT & PERIOD COVE
MS Thesis

6. PERFORMING ORG. REPORT NUMB

7. AUTHOR(s)
Donald W. Gross
Captain

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Air Force Institute of Technology (AFIT-EN)
Wright-Patterson AFB, Ohio 45433

10. PROGRAM ELEMENT, PROJECT, T
AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

12. REPORT DATE
December, 1982

13. NUMBER OF PAGES
81

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADI
SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release, IAW 190

Approved for public release: IAW AFR 190-17.

Lynn E. Wolaver
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (ATC)
~~Wright-Patterson AFB OH 45433~~

4 JAN 1983

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Estimation
Bayes filter
Parameter identification

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A technique was developed for estimation of launch vehicle perfc mance parameters. This technique used an inverse covariance or Bayes filter. Both a seven state and an eight state dynamics mo were implemented and their performance investigated. Observatio consisted of angular, infrared measurements from two orbital sensors. The seven state filter had 3 position, 3 velocity and an acceleration component for its state vector. The acceleration state was modelled as constant between measurement updates. Aft the addition of a fading memory, the seven state filter showed good performance in estimating a variable acceleration profile. The eight state filter had 3 states each for position and velocity, and seventh and eighth states involving engine exit velocity and propellant mass flow rate. Although the eight state filter had a better model for the acceleration, the filter proved to be unsuccessful in its estimation attempts.